GAMMA 10

PROGRAMMERS' REFERENCE MANUAL

DE LA RUE BULL Machines Limited
114 - 118 Southampton Row LONDON WC1 Tel.CHAncery 5725

No.

This manual is the property of
De La Rue Bull Machines Limited.
It is not transferable and may not
be sold.

LIST OF CONTENTS

CHAPTER ONE

GENERAL DESCRIPTION (Pages 3 and 4)

CHAPTER TWO

FUNCTIONAL DESCRIPTION

Para

CARD READER/PUNCH

GENERAL 1
DESCRIPTION 2

OPERATION

Reading.. 5
Reading checks 9
Punching checks 10

PRINTER

GENERAL 11
DESCRIPTION 13
OPERATION.. 14
Paper Control Tapes 17
Paper movement safeguards 22
Printing.. 23
Printing checks 24
CENTRAL PROCESSOR 27
HIGH SPEED STORE 29

Para

Data In The High Speed Store 30

Areas 32

Markers 36

Programme in the High Speed Store 46

HSS Capacity Available for Data and Programme 50

ARITHMETIC UNIT 51

CONTROL UNIT 52

BUFFER STORES 53

Input Buffer 57

Condition Buffer 58

Punch Buffer 61

Print Buffer 64

CONDITION PLUGBOARD 66

EXAMPLE 1 - Detecting card types at RS2.. 68

EXAMPLE 2 - Detecting changes of card group. 71

EXAMPLE 3 - Detecting the last card in a file 74

WIRING RESTRICTION 75

CHAPTER THREE

PROGRAMMING THE GAMMA 10

GENERAL 1

PROGRAMME INSTRUCTIONS.. 4

Page

INPUT INSTRUCTIONS 26

RAC - Read A Card 27

INC - Input Numeric Characters 28

Page

IAC - Input All Characters 29

OUTPUT INSTRUCTIONS 30

PRI - Print (and paper advance) 31

SSP - Select Stacker Punch 32

OAC - Output All Characters 33

ONU - Output Numeric Unsuppressed 34

ONS - Output Numeric Suppressed 35

OSC - Output Single Character 36

TRANSFER INSTRUCTIONS 37

TRF - Transfer 38

TF1 (2) (3) - Transfer from Group (1) (2) (3) 39

TT1 (2) (3) - Transfer To Group (1) (2) (3) 40

MARKER INSTRUCTIONS 41

TCC - Transfer Condition to Condition 42

ASC - Allocate Sign according to Condition 43

ACS - Allocate Condition according to Sign 44

TSS - Transfer Sign to Sign 45

ARITHMETIC INSTRUCTIONS 46

ADD - Add 47

SUB - Subtract 48

MUL - Multiply 49

DIV - Divide 50

CMP - Compare 52

JUMP INSTRUCTIONS 53

JI= - Jump Instruction = ... 54

JIU - Jump Instruction Unconditional 55

JID - Jump Instruction Dummy 56

JIS - Jump Instruction SET 57

Page

JIN - Jump Instruction NONSET 57

JS= - Jump Segment 58

JSU - Jump Segment Unconditional 59

JSD - Jump Segment Dummy 60

JSS - Jump Segment SET 61

JSN - Jump Segment NONSET 61

SUBROUTINE RETURN INSTRUCTIONS 62

SRA - Store Return Address 63

JBK - Jump Back 64

SHIFT INSTRUCTIONS 65

TWA - Transfer Within Area 66

SHR - Shift Right 67

SHL - Shift Left 68

SCF - Shift Character Fill 69

MISCELLANEOUS INSTRUCTIONS 70

MOD - Modify 71

HLT - Halt 72

IBC - Insert Blank Card 73

EMP - Empty 74

CHAPTER FOUR

CONTROL PANELS (Pages 75 to 80)

CHAPTER FIVE

MANUAL OPERATIONS

Para

SETTING UP PROCEDURES

POSITIONING THE CONDITION PLUGBOARD 1

SWITCHING ON.. 2

PRINTER 3

Paper Control Tape 4

Continuous Stationery 10

CARD READER/PUNCH 15

STARTING A JOB 16

ALLOCATING HSS AREAS AND PROGRAMME LOADING 17

LOADING THE AREAS AND THE MARKERS.. 20

SUPPRESSING FUNCTIONS 21

MARKER PUSH-BUTTONS 22

STARTING PROCESSING 23

CHAPTER SIX

INCIDENTS AND RESTART PROCEDURES

INCIDENTS DURING PROGRAMME LOADING AND TESTING.. 2

INCIDENTS DURING PROCESSING

Red Lamp (Non-Labelled) Glows 3

Incidents On Card Reader/Punch

Read Check Incident 4
Punch Check Incident 5
Card Jam On Primary Track 6
Card Jam On Secondary Track 7

Para
Card Jam On Common Track 8
Card Slot Jam On Primary Track 9
Card Slot Jam On Secondary Track 11
Full Stacker 13

Incidents On The Printer

End Of Paper (Or Torn Paper).. 14
Faulty Paper Movement (Abnormal Skip Or Torn Paper) 15
Ribbon Fault 16

Programme Faults

Capacity Overflow 17

STOPS DURING PROCESSING

STOP PUSH-BUTTON 18

PROGRAMMED STOP 19

SWITCHING OFF 20

LIST OF APPENDICES

APPENDIX

1 - Physical Characteristics Of The Gamma 10.

2 - Gamma 10 Punch And Machine Codes.

3 - Instruction Codes

4 - Data Distribution in The High Speed Store

5 - Typical Programme Input Sequences

6 - Typical Programme Output Sequence for Punching

7 - Typical Programme Output Sequence for Printing

8 - Card Formats

9 - Condition Plugboard

INTRODUCTION

The GAMMA 10 is a punched card computer providing many of the advantages usually associated only with much larger and more complex systems.

Simultaneity of reading, processing, printing and punching at a high speed is achieved through the advanced design of the Central Processor.

The capacity of the High Speed Store coupled with the brevity and power of the computer instructions allows a complex programme to be stored and large volumes of data to be processed. Arithmetic operations are carried out by a sub-unit brought into operation by a single instruction.

The detection of particular cards and changes of card groups is greatly simplified through the use of the Condition Plugboard which requires the minimum of wiring.

Controlled buffer stores allow the programmer to ensure that only the data to be processed is introduced into the High Speed Store thus avoiding wastage of time and storage area.

From the advanced design of the GAMMA 10 computer, a number of important operational advantages are derived:

- Productivity is increased by the efficiency of the GAMMA 10. Card consumption is appreciably reduced.
- During processing, the Card Reader/Punch can automatically update card files - a feature usually only available with much larger systems.
- The programmer benefits from the advantages of writing programmes in a simple symbolic code which does not need a separate compiling routine.
- The programmer is freed from the problems of deriving addresses. Similar indirect addressing systems are used for both the programme and the data.
- Programme segmentation simplifies the modification and testing of programmes and allows standard subroutines and standard programmes to be used, so reducing programming time.

Personnel training is reduced to a minimum. The transition time from any existing system to data processing using the GAMMA 10 is minimized since in many cases the existing punched card files can be used.

The GAMMA 10 requires no special environmental conditions - its space requirements are minimal.

CHAPTER ONE

GENERAL DESCRIPTION

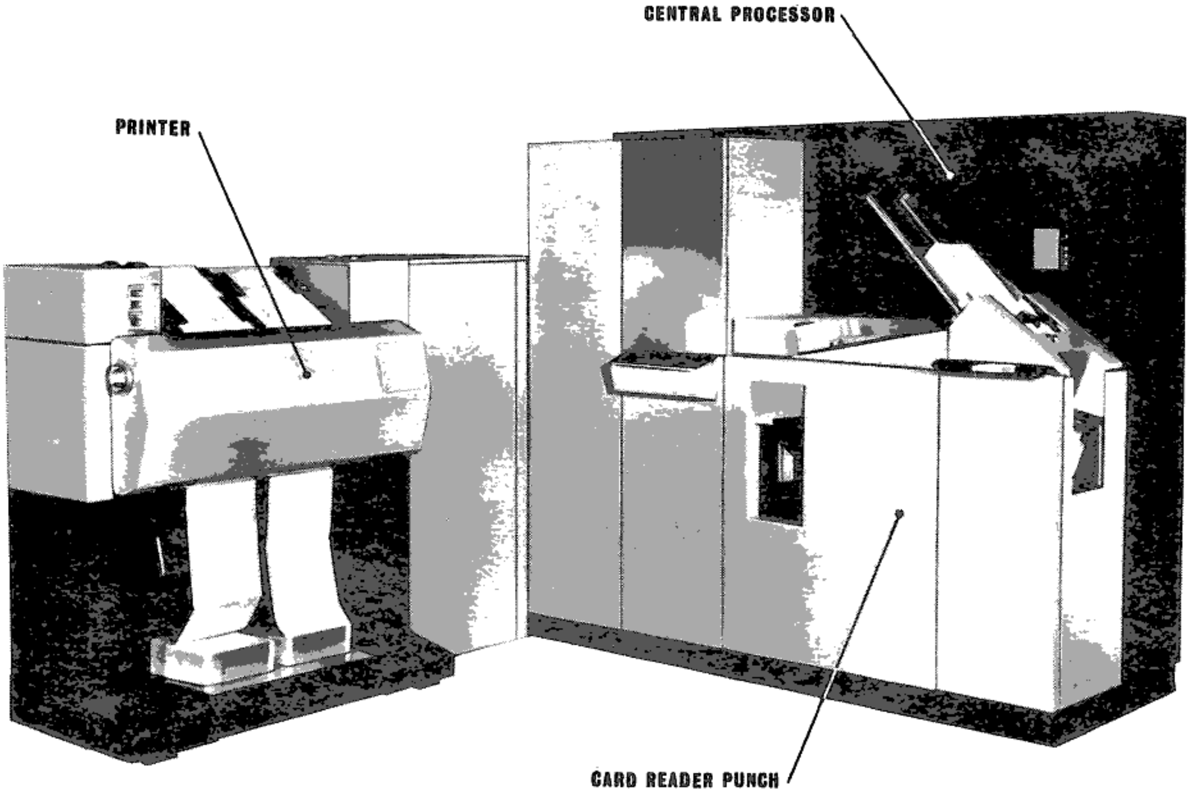


Fig.1: General view of Gamma 10

1. Figure 1 is a general view of the Gamma 10 showing its main functional units. Appendix 1 gives the physical characteristics (dimensions, weight etc.) of the computer.

2. The CARD READER/PUNCH reads data into the Central Processor and can also punch cards with data extracted from the Central Processor.

3. The PRINTER produces printed statements containing data extracted from the Central Processor.

4. The CENTRAL PROCESSOR contains:

(1) The High Speed Store which is used to store the programme and the data to be processed.

(2) The Control Unit which co-ordinates the operation of the input/output units and the programme.

(3) The Arithmetic Unit which executes the arithmetic operations.

(4) Buffer stores which link the input/output units to the High Speed Store to ensure simultaneity of input, output and processing.

5. These units and their controls are arranged so that the operator can supervise their operation from a central position. A control console, on the front of the Central Processor, provides the operator with switches and indicator lamps which he uses to control the operation of the computer.

CHAPTER TWO

FUNCTIONAL DESCRIPTION

CARD READER/PUNCH

GENERAL

1. The Gamma 10 Card Reader/Punch operates under programme control on 80-column cards at 300 cards/minute. It can read data from a card and either punch the results of calculations on this data into the same card, into a blank card or into a following card. Constants can be reproduced. Cards pass normally into the normal stacker or they may be selected into one of two special stackers. An automatic accuracy check is made on all data read and punched.

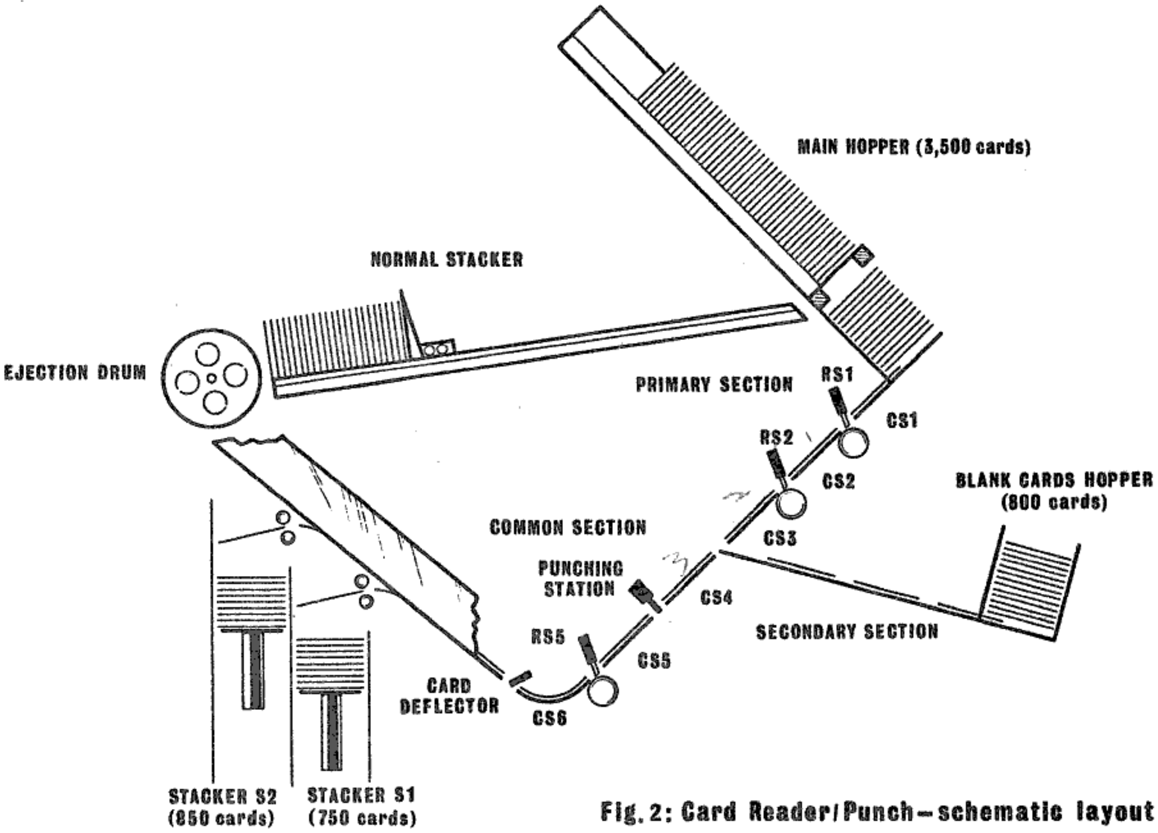


Fig. 2: Card Reader/Punch—schematic layout

DESCRIPTION

2. Figure 2 is schematic representation of the Card Reader/Punch showing the card tracks, hoppers, stackers, reading stations and the punching station.

3. Cards are fed from the Main hopper (3, 500 cards capacity with constant automatic jogging of the bottom 500 cards) onto the Primary section of the main track where they are read by Reading Station 1 (RS1), and one cycle later, by Reading Station 2 (RS2). The cards

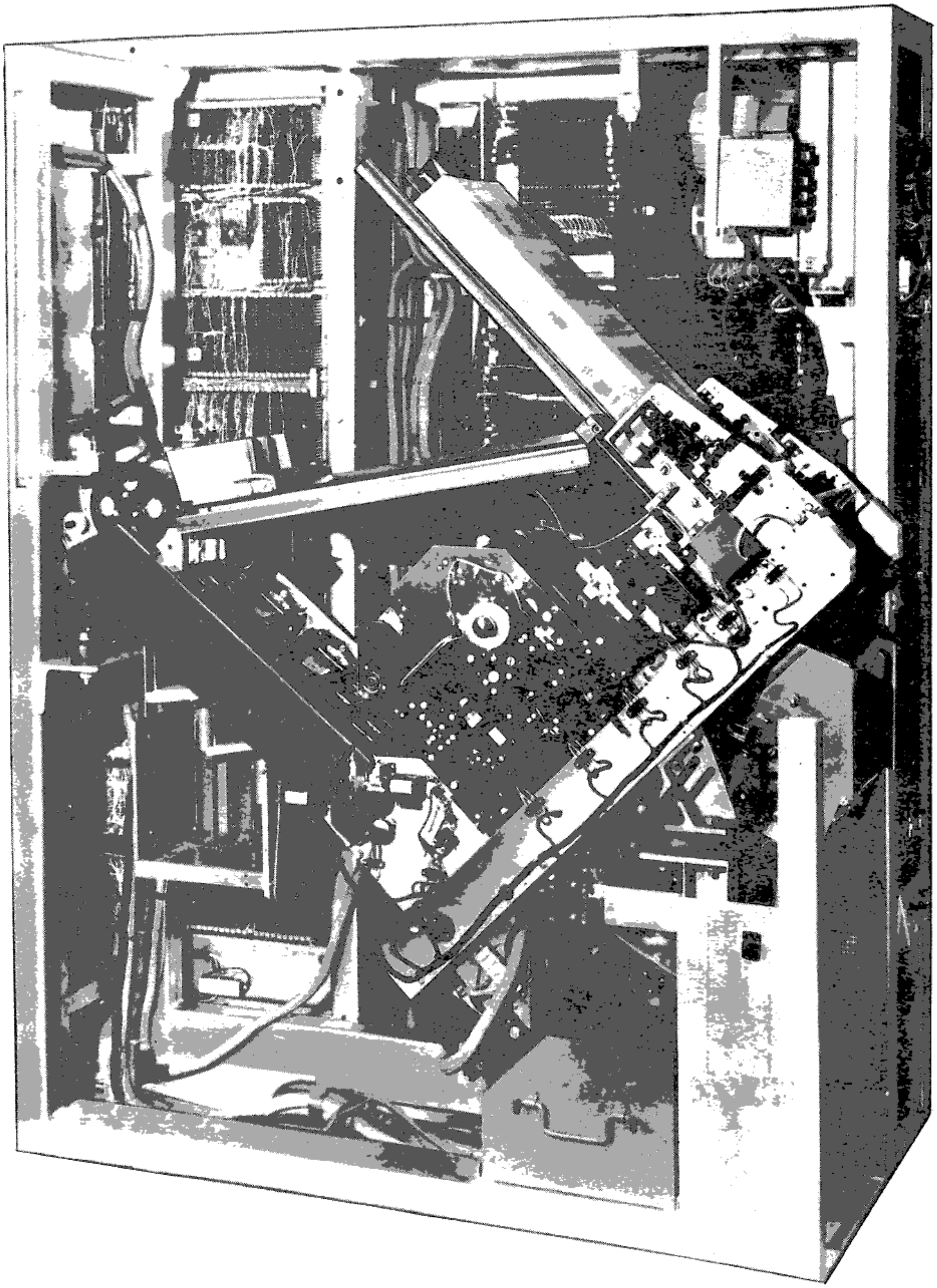


Fig.3: Interior of Card Reader Punch

are then passed to the Common section of the track (Fig. 2) where they may be punched and where they are read by Reading Station 5 (RS5) to check any punching which may have been carried out as they passed the Punching Station. Finally, cards may be selected into one of the two special stackers, or they may simply pass into the normal stacker.

4. The Secondary track joins the main track before the Punching Station, to allow blank cards on the secondary track to be inserted and punched with the results of data processed from one card or from a group of cards. Feeding by the primary section of the main track is delayed until this operation is completed. In all cases however, the track is capable of feeding 300 cards a minute.

Note... card stations on the main track are numbered CS1 to CS6. Card stations on the secondary track are numbered CS1', CS2', CS3'.

OPERATION

Reading

5. One mechanical cycle of the Card Reader/Punch lasts 200 milliseconds (300 cards/minute). Of this time, 160 milliseconds are required to read a card. The remaining 40 milliseconds elapse between reading the last possible punch (punch 12 - cards fed 9 edge leading) and the first possible punch on the following card. Figure 4 is a simple timing diagram illustrating this point.

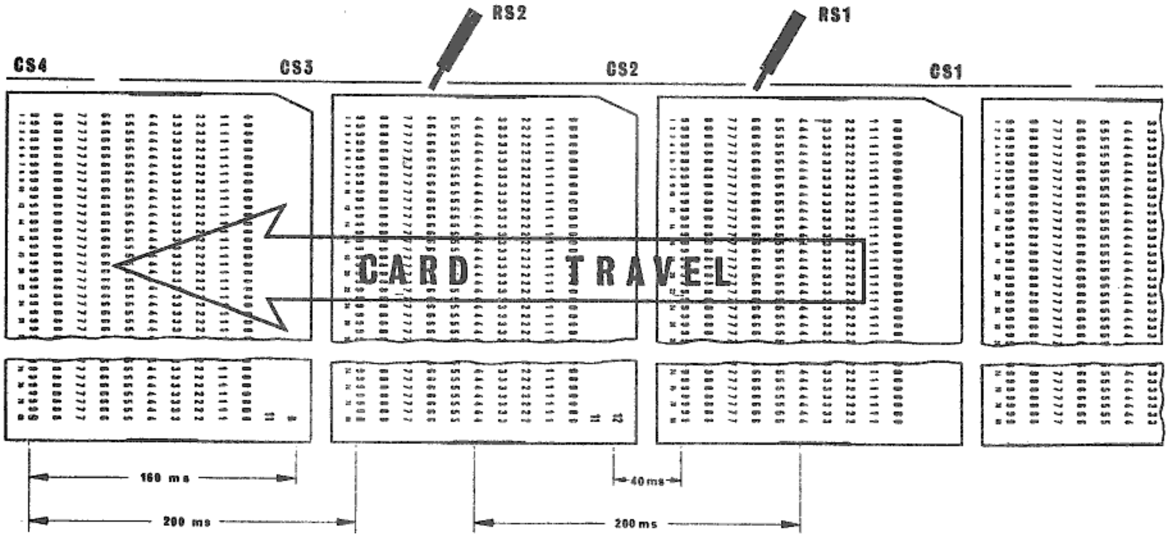


Fig.4: Card Reading Times

6. From the diagram it can also be seen that when a given row on a card is being read at RS2, the corresponding row on the following card is also being read at RS1.

7. Reading stations 1 and 2 are linked to the Condition Plugboard and, via wiring on the Condition Plugboard, to the Condition buffer. The Condition Plugboard is used to identify cards and information - a full description of it is given in pages 19 to 22.

8. Thus, data read from cards as they pass RS1, is used to identify cards and information and can only be loaded into the Condition buffer, while data read from cards as they pass RS2 is used for the same purpose but can also be loaded into the Input buffer.

Reading checks

9. For every card read at RS1 a residue check digit is calculated on the number of punches in the card. A check digit is calculated similarly for every card read at RS2. The check digits calculated at RS1 and RS2 are compared, and should any difference be detected, the machine stops before the suspect data is trasferred to the HSS. Lamps on the Control Console indicate this situation (Chapter 4).

Punching checks

10. The check digit calculated for a card as it passes RS2 is stored and modified in accordance with the content of the Punch buffer to be punched in the card. When the same card passes RS5, a second residue check digit is calculated, on the number of punches in the card, and compared with the modified RS2 check digit. Should any difference be detected, the computer stops and the suspect card is ejected into S1. Lamps on the Control Console indicate this situation (Chapter 4).

PRINTER

GENERAL

11. The Printer operates on-line at 300 lines/minute. The 120-position print drum has fifty-nine characters engraved on its circumference at each of the 120 print positions. The characters are spaced at 8 to the inch and printed lines are spaced at 6 to the inch.

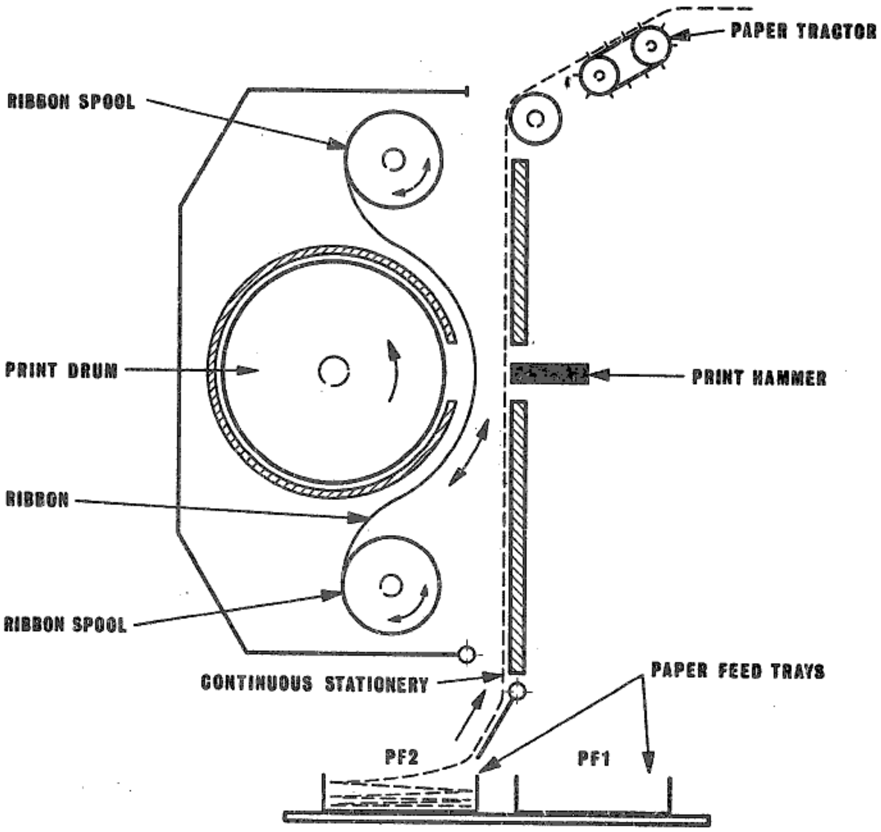


Fig. 5: Printer—diagrammatic cross-section

12. One paper feed, Paper Feed 1 (PF1) is fitted as standard to every Gamma 10 Printer. A second paper feed (PF2) can be fitted to allow two different statements of the same width or of different widths to be printed side by side. With the two paper feeds fitted, each is independently controlled.

DESCRIPTION

13. Figure 5 is a diagrammatic cross-section of the Printer showing stationery being fed from a tray at the front of the Printer and passing upwards between the print drum and print hammers into a tray at the rear of the Printer. The continuous stationery shown in the diagramm is being fed on PF2 - the stationery on PF1 is fed from the rear tray.

OPERATION

14. Printing is effected by the print hammers which press the paper onto a carbon ribbon interposed between the paper and the print drum.

15. The carbon ribbon is common to the two paper feeds. It has a life of approximately 1,400,000 printed lines. The ribbon winds and unwinds on two spools, one above and one below the print drum. Ribbon reversal is automatic.

16. The paper feed mechanisms are above the print drum. The paper is drawn upwards through the Printer at about 16 in/s by tractors which engage sprocket holes in the margins on either side of the paper. The tractors can be moved laterally and relative to each other to allow variation in the positioning of the paper relative to the print drum and various widths of stationery to be used. Manual controls are provided for adjustment of the tension of the paper and of the exact setting of synchronism between the paper and its control tape.

Paper Control Tapes

17. The paper control tapes move in synchronism with the paper under the lefthand cover of the Printer (Fig. 6).

18. Each tape is joined in a continuous loop so that every sheet on the same paper feed is controlled by the same tape programme. They are 8-track tapes with feed holes running the length of the tape between tracks 4 and 5. Each feed hole corresponds to one line-space on a sheet which must be a multiple of three lines long. The number of feed holes must be a multiple of three between 60 and 120. Thus, for stationery with 30-line sheets, the corresponding tape would have to have 60 feed holes to cover two complete sheets.

19. The various end-of-skip points are punched in the eight tracks (numbered 0 to 7) of the tape. The tracks are used as follows:

(1) Tracks 0 to 5 command six end-of-skip points as required by the user.

(2) Track 6 is used to stop a skip at the last printing line on a sheet.

(3) Track 7 is used for the Completed Form (CF) function.

20. The movement of the paper always takes place before the printing action. By programme (PRI instruction) the paper can be moved 0, 1, 2 or 3 line-spaces otherwise it is always controlled by its corresponding control tape.

21. A skip is stopped either by a count down on a line-space counter or, when a punch is detected in one of the tracks (0 to 6) of the control tape.

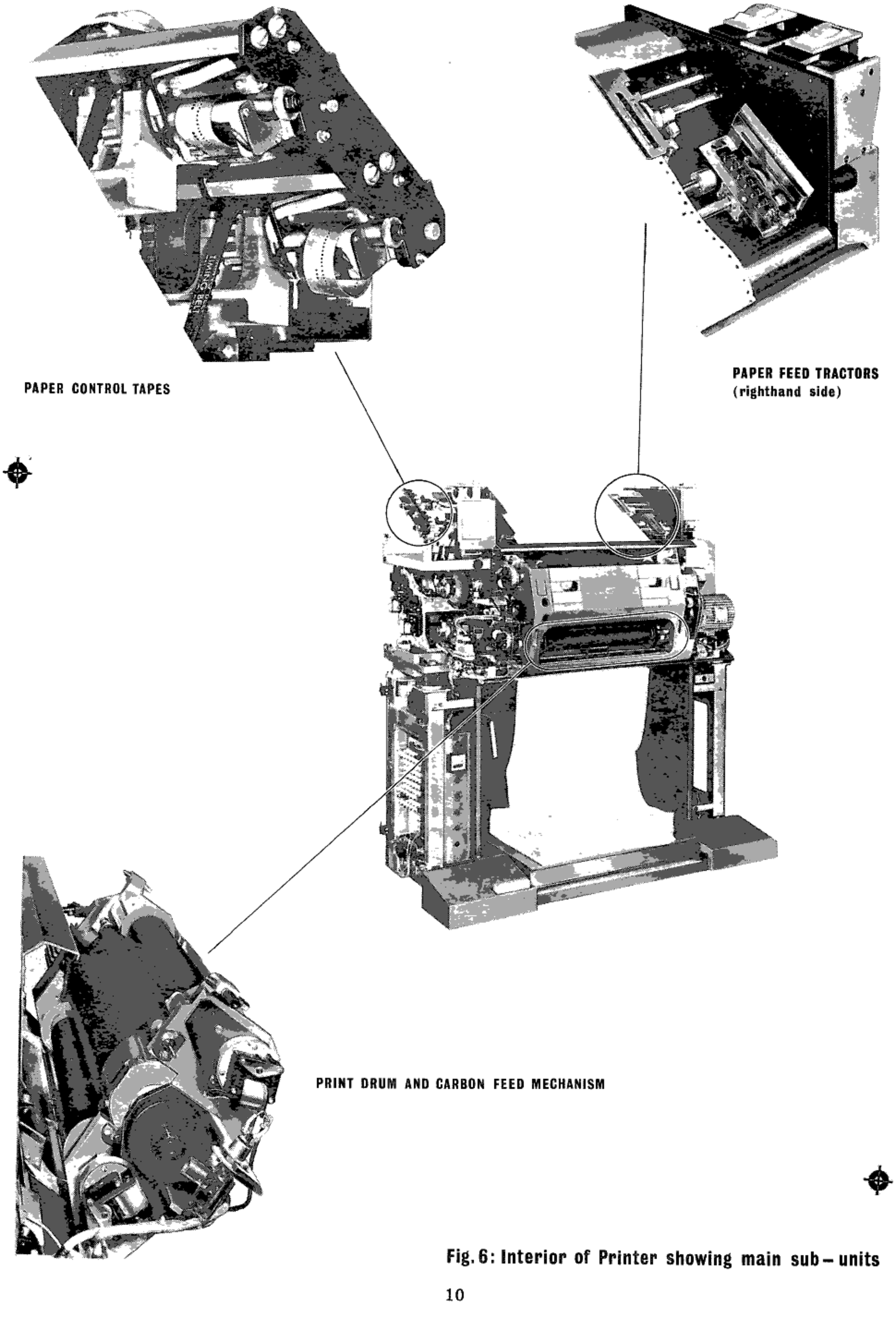

PAPER CONTROL TAPES
PAPER FEED TRACTORS (righthand side)
PRINT DRUM AND CARBON FEED MECHANISM
TIMING BELT

Fig. 6: Interior of Printer showing main sub-units

Paper movement safeguards

22. Various safeguards operate during printing. The possible malfunctions, their effects and remedies are described in Chapter 6.

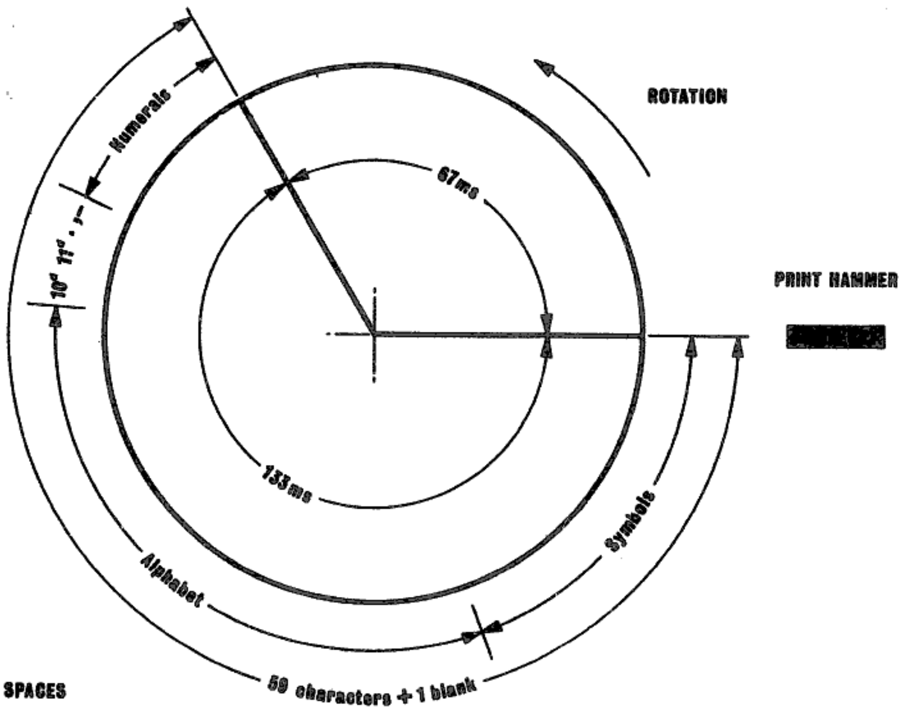
Printing

23. The print drum rotates continuously at 300 rev/min. When a row of characters, for example M, is in the printing position, the Print buffer is scanned to determine in which positions an M is to be printed. The scan consists of a series of 120 comparisons, of the characters stored in the Print buffer with characters (in this case M) generated by an optical character generator rotating in synchronism with the print drum. When the result of the comparison is parity the corresponding print hammer for that position is selected and the content of the corresponding location in the Print buffer is cleared. Printing occurs when the entire 120 comparisons have been carried out. Printing is finished when the entire 120 locations of the Print buffer are cleared. Printing of a line can begin as soon as the Print buffer is loaded and the paper movement, which precedes printing, is finished.

Printing checks

24. As described above, as a character is printed, its location in the Print buffer is cleared. Thus, when all the fifty-nine characters have passed the printing position the entire 120 locations of the Print buffer should be clear. If this is not the case, the computer stops and the red lamp, on the Control Console, lights up.

Length of Skips



1 REVOLUTION: 200 ms
" " " " : 16 LINE SPACES
BLANK SECTOR: 5 LINE SPACES APPROX.

Fig.7: Print Drum Timing

25. Figure 7 shows the arrangement of the characters on the print drum and the time taken by the characters to pass the print hammers. The figure also shows that if the last character printed, in a given line, is the last numeral to appear in front of the print hammers, only five lines can be skipped if the first character to be printed in the next line is the first symbol to appear in front of the print hammers. This, however, is the worst possible case. The maximum length of skip possible between two consecutive printing lines n and n+1 when printing at 300 lines/minute is given by the formula:

$s = (d-3)\ /5$ rounded down to the nearest whole number.

where: s is the maximum number of line-spaces possible in the skip and d is the number of characters or character positions on the print drum between the last character printed in line n and the first character to be printed in line n+1.

26. The following table illustrates the results normally obtained.

	Number of characters in line n+1			
Number of characters in line n	59	10 numerals 26 letters 10 11 , - .	10 numerals 10 11 , - .	10 numerals
18 symbols 26 letters	s = 8	s = 12	s = 17	s = 18
18 symbols 26 letters 10 11 , - .	s = 7	s = 11	s = 16	s = 17
59	s = 5	s = 9	s = 14	s = 15

Table 1: Lengths of skips

CENTRAL PROCESSOR

27. The Central processor is the main functional unit of the Gamma 10. It controls and co-ordinates the operation of the input/output units and the processing units which it contains. Within the Central Processor are:

(1) The High Speed Store (HSS) which stores the programme and the data to be processed.

(2) Buffer stores which link the HSS to the input/output units to ensure simultaneity of input, output and processing.

(3) The Arithmetic Unit which executes the arithmetic operations.

(4) The Control Unit which co-ordinates the operation of the input/output units and the programme.

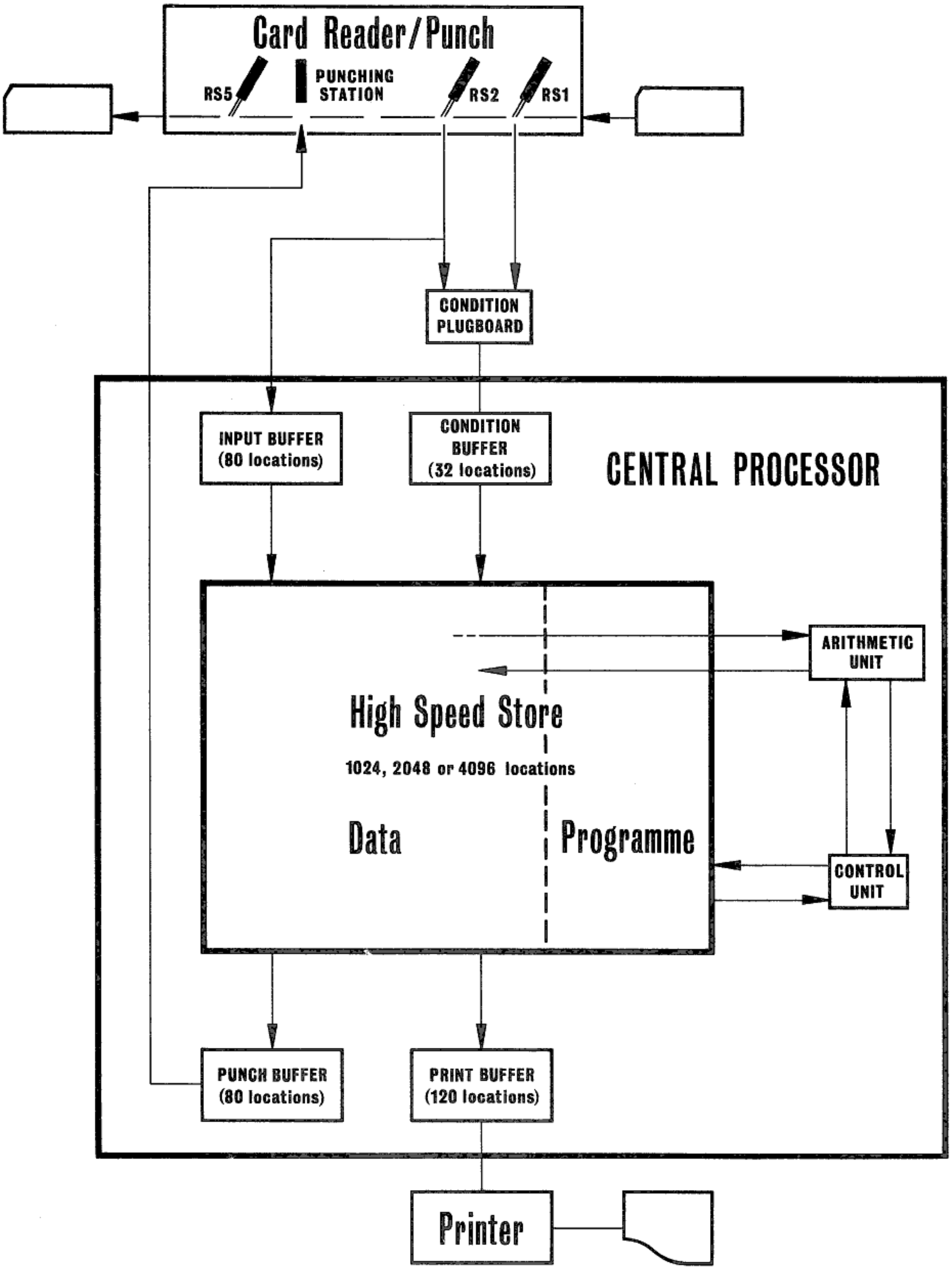

Card Reader/Punch
RS5
PUNCHING STATION
RS2
RS1
CONDITION PLUGBOARD
INPUT BUFFER (80 locations)
CONDITION BUFFER (32 locations)
CENTRAL PROCESSOR
ARITHMETIC UNIT
High Speed Store
1024, 2048 or 4096 locations
Data
Programme
CONTROL UNIT
PUNCH BUFFER (80 locations)
PRINT BUFFER (120 locations)
Printer

Fig. 8: Functional layout of Gamma 10

28. Figure 8 illustrates the functional layout of the computer and shows how its units are interconnected.

HIGH SPEED STORE

29. The principal characteristics of the HSS are

(1) Its cycle time of 7 microseconds.

(2) Its capacity of 1024, 2048 or 4096 character locations each of which can be used to store data or programme.

(3) It is a character store i.e. every location can be used individually. Characters are transferred one by one to and from the HSS. Character stores have the advantage that data can be loaded into store without wasting storage area. In word stores, in which the storage area is previously and permanently divided into words of a fixed number of characters, storage area is frequently wasted because data is naturally variable in its storage requirements and so is rarely divisible among a whole number of word areas.

Data In The High Speed Store

30. Each High Speed Store location is made up of seven magnetic cores six of which are used to store the binary code of a Gamma 10 character. With six magnetic cores each having two possible states (binary 0, and binary 1) 64 specific combinations can be obtained of which sixty are used to represent coded data. For example the letter A is coded:

0 1 1 0 1 0

31. The seventh magnetic core in each location is used to store the check bit for each character. The check bit is an integral part of a stored character and remains with the character coding as long as the character is in the High Speed Store. Odd parity checking is used in the Gamma 10. This means that the sum of the 1 bits within every character must always be odd. Thus, since the binary code of the letter A has three 1 bits the complete code of A stored in a High Speed Store location is:

$\boxed{0}$ 0 1 1 0 1 0

On the other hand, the letter B coded:

0 1 0 1 1 1

has four 1 bits and the odd parity bit must therefore b 1 to give:

$\boxed{1}$ 0 1 0 1 1 1

Areas

32. HSS locations which are used to store data are divided into areas, each of which is referred to by an area number. The programmer fixes the lengths (in locations) of the areas according to the requirements (number of characters and spaces) of the data which is to be stored, and allocates the area numbers as required. This is illustrated in the table opposite.

33. Every Gamma 10 programme can use up to sixty-four areas (numbered 00 to 63) of which areas 01 to 63 can be allocated any length from 0 to 63 locations. This group of areas is known as Group 0 and area 00 in this group always has length 0 and is therefore unavailable

for storing data.

DATA		AREA	
TYPE	LENGTH (Characters)	LENGTH (Locations)	NUMBER
Customer Name	20	20	01
Town	16	16	02
Account Number	7	7	03

34. Group 0 is the only group available in Gamma 10 computers which have a HSS with 1024 locations. However, one additional group (Group 1) can be created if a Gamma 10 has a HSS with 2048 locations, and up to three additional groups (Group 1, Group 2 and Group 3) can be created if a Gamma 10 has a HSS with 4096 locations. Areas in groups 1, 2 and 3 have their lengths allocated by the programmer in exactly the same way as areas in Group 0 but area 00 in these groups can have any length up to the maximum of 63 locations.

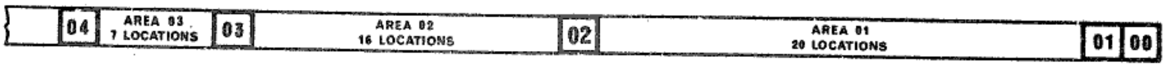
35. Data stored in a Group 1 or Group 2 or Group 3 area cannot be processed until it has been transferred to an area in Group 0. During the course of a job, data can only be loaded into an area in Group 1 or Group 2 or Group 3 from a Group 0 area. During programme loading, however, areas in Groups 1, 2 and 3 may be loaded with data punched in an Area Card (Appendix 8).

Markers

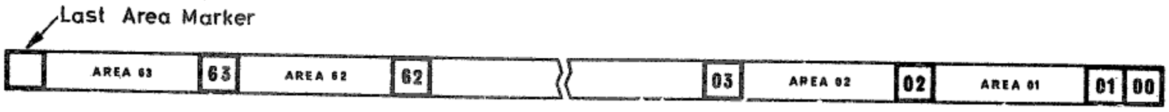
36. Within the HSS, areas are stored in their numeric sequence. The limits (beginnings and ends) of the areas are defined by markers. Figure 9 shows how the data in the table above would be stored and how markers (numbered 00 to 63) are positioned relative to areas also numbered 00 to 63.

37. Every marker occupies one HSS location. Markers are loaded into the HSS from Area Cards (Appendix 8) which define the lengths of the areas and the group to which the area belongs. For Group 0, sixty-four markers are always loaded even when all of the Group 0 areas have not been allocated specific lengths. For areas in Group 1, Group 2 and Group 3, markers are loaded only when areas in these groups have been allocated specific lengths. A minimum of 64 HSS locations are therefore unavailable for storing Quantitative data (as differentiated from Qualitative data e.g. codes, arithmetic signs etc.) because they are used for markers. Qualitative data, can however be stored in a marker as described in paragraphs 41 to 45 and 66 to 73.

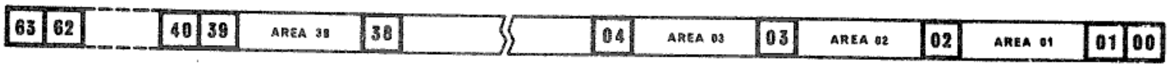
38. A final marker must be loaded to define the end of the last area in every group. This marker is loaded from the last Area Card. The first area in any group always has its beginning defined by marker 00 for its group. This is illustrated in Figure 9:



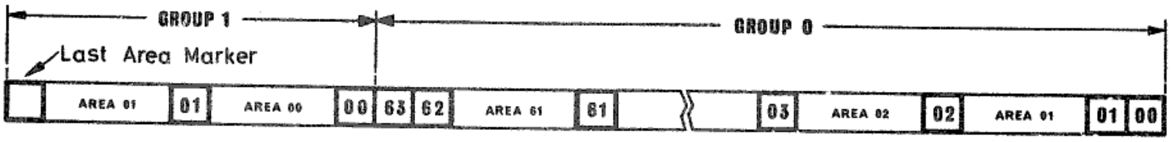
GENERAL CASE FOR ANY GROUP



GROUP 0 ONLY USED – ALL AREAS ALLOCATED



GROUP 0 ONLY USED – 38 AREAS ALLOCATED



GROUP 0 AND GROUP 1 USED

NOTE: AREA 00 IN GROUP 0 ALWAYS
HAS LENGTH 0

Fig. 9: Positioning of markers

39. The programme uses the HSS addresses of the markers to locate the data stored in the areas. For this purpose, the HSS address of every marker is stored in a two-location Area Address Register (AAR). In the foregoing example:

HSS address of marker 00 is stored in AAR 00
HSS address of marker 01 is stored in AAR 01
HSS address of marker 02 is stored in AAR 02
HSS address of marker 03 is stored in AAR 03

40. For every group of areas there are therefore 64 Area Address Registers occupying a total of 128 locations.

41. Four combinations of the binary states of the seven cores which make up the location in which a marker is stored are used to represent Qualitative data which the computer recognises as sign POSITIVE, sign NEGATIVE, condition SET, condition NONSET. The codes are shown in the table below:

CODE							SIGN	CONDITION
1	0	1	1	0	1	1	+	NONSET
0	0	1	1	1	1	1	+	SET
0	1	1	1	0	1	1	-	NONSET
1	1	1	1	1	1	1	-	SET

42. The sign stored in a marker applies to the data stored in the area to which the marker refers.

43. The condition stored in a marker may refer to data in any area, determined by the programmer, or to some aspect of the programme. Examples of the applications of SET and NONSET conditions are given in pages 21, 22 and 23.

44. There are sixty-four markers (00 to 63) in Group 0. Marker 00 in this group is always SET and POSITIVE and is used in programme instructions requiring signs and conditions of other markers to be changed (see TCC, ASC, ACS, TSS instructions pages 42, 43, 44 and 45 respectively). The initial signs and conditions of Group 0 markers are determined by the programmer and are loaded from the Area Cards (Appendix 8). The conditions in markers 1 to 32 can be controlled from the Condition Plugboard (page 19). The conditions in markers 61, 62 and 63 can be altered from the Control Console (Chapter 4 page 78). Markers 59 and 60 take the SET condition when the Completed Form (CF) state is detected on PF2 and PF1 respectively. Signs and conditions can be transferred from one Group 0 marker to another Group 0 marker by the programme.

45. Markers defining areas in Group 1, Group 2 and Group 3 are not numbered, they can not store conditions but they do store the sign of the area whose beginning they define. The number of markers used to define these areas depends upon how many areas in these groups have been allocated lengths (para 37).

Programme in the High Speed Store

46. The programme of the Gamma 10 is stored in the HSS. It is made up of programme segments, of which there may be sixty (numbered 01 to 60), each of which may consist of up to sixty-four instructions (numbered 00 to 63).

47. The programme is stored in segment number order immediately preceding the area allocated to the data to be processed. This is shown in Figure 10 in which the letters O, A, B represent the three characters used to define a Gamma 10 instruction (Chapter 3 para 4) and occupy one location each in the HSS.

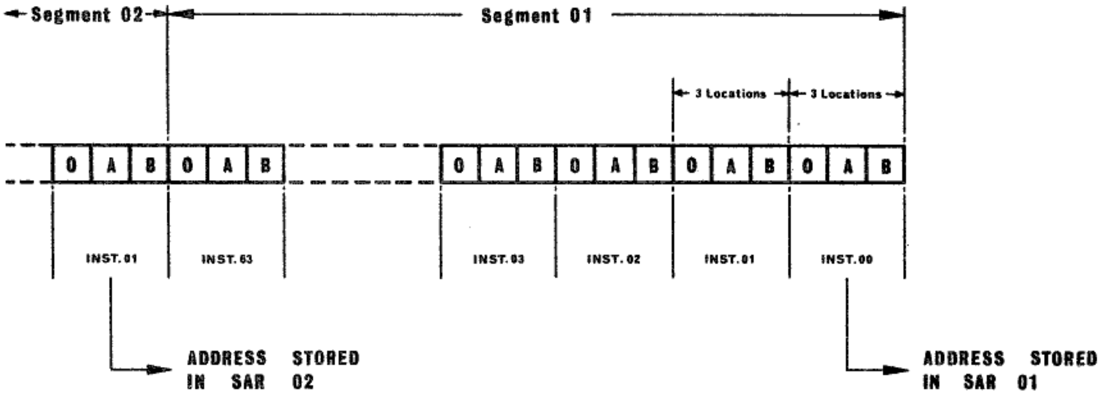


Fig.10: Programme in the High Speed Store

48. The number of the first instruction of every segment and its segment number are automatically stored in a two-location Segment Address Register (SAR) when the programme is loaded. These registers provide the programme with references when jumps from segment to segment occur.

49. There are sixty-four Segment Address Registers (00 to 63) occupying a total of 128 locations. SAR 00 has a special function in segment jump instructions (pages 58 to 61). SAR 63

is used in programme testing and SAR 61, and SAR 62 perform specific programme functions which can not be controlled by the programmer.

HSS Capacity Available for Data and Programme

50. From the foregoing it is possible to calculate the capacity of the HSS which remains available to store the data and the programme when the locations occupied by markers, Areas, Area Address Registers and Segment Address Registers have been totalled and deducted from the overall capacity of the HSS. The only additional locations to be deducted are 24 which are reserved for servicing purposes. A summary of these figures is given in Appendix 4.

ARITHMETIC UNIT

51. This unit performs the four arithmetic operations Add, Subtract, Multiply, Divide. It operates algebraically, controlled by the Control Unit. Data to be processed by this unit must be stored in Group 0 areas when the arithmetic operation is to be executed.

CONTROL UNIT

52. This unit co-ordinates the operation of the functional units of the computer during the execution of the programme.

BUFFER STORES

53. The buffer stores act as intermediaries in data transfers between the HSS and the input/output units. They have magnetic cores, with the same cycle time as the HSS, as their storage medium, and provide simultaneity of input, processing and output by ensuring that optimum use can be made of the HSS while the slower input/output units are carrying out their functions. This latter function is explained below taking as an example the Input buffer store.

54. Data read from a card in 160 milliseconds (40 ms elapse between cards) is loaded into the Input buffer and the Condition buffer without any effect on the HSS. When reading is finished, the data can be transferred from the Input buffer and the Condition buffer into the High Speed Store. This transfer takes a very short time, less than 1 millisecond, and the HSS is then free to carry out processing while the next card is being read.

55. The Print and Punch buffer stores play an identical part in ensuring maximum processing time by storing data which is to be punched or printed while further output data is being produced in the HSS.

56. Thus, the provision of buffer stores in the Gamma 10 ensures complete simultaneity of reading, processing, punching and printing. While the information from one card is being processed, the succeeding card is being read and the preceding card may be punched. Printing may also take place at the same time.

Input Buffer

57. The Input buffer has 80 locations corresponding to the 80 columns on a punch card. Data read on the Card Reader/Punch at RS2 is loaded directly into the Input buffer. As the data is loaded it is automatically transcoded to enable it to be stored in the magnetic cores. Simultaneously with the transcoding parity is checked and odd parity bits allocated as required.

Condition Buffer

58. The Condition buffer has 32 one-bit locations which correspond to markers 1 to 32.

59. Reading impulses from RS1 and RS2 on the Card Reader/Punch are transmitted to the Condition Buffer by the Condition Plugboard. Every location which receives an impulse stores binary 1. Every location which does not receive an impulse stores binary 0 i.e. the state to which every location is restored at the end of every cycle.

60. When a card is read, the content of the 32 locations of this buffer are transferred to their corresponding markers in the HSS. When this transfer takes place:

(1) Every marker which receives binary 1 from the buffer takes the condition SET.

(2) Every marker which receives binary 0 from the buffer takes the condition NONSET.

(3) No changes of sign take place in markers 1 to 32.

(4) Every buffer location is restored to binary 0.

Punch Buffer

61. The 80-location Punch buffer stores data, extracted from the HSS, which is to be punched in cards as they pass the Punching Station on the Card Reader/Punch.

62. The cards are punched row by row. As the cards are punched data in the buffer is automatically transcoded to enable the cards to be punched in accordance with the punch code for which the computer was built.

63. The buffer locations are numbered (1 to 80) to enable the correct correlation between the data stored in a location and the card column in which it is to be punched to be maintained. The content of a buffer location is always punched in the card column of the same number. Distribution of data to accord with the format of the card in which it is to be punched must be effected by the output instructions which transfer the data from the HSS to the buffer. Examples of how this is done are shown in Appendices 6(i) and 6(ii).

Print Buffer

64. The 120-location Print buffer stores data, extracted from the HSS, which is to be printed.

65. The buffer locations are numbered 1 to 120 to correspond with print positions 1 to 120 (numbered from left to right). The content of a buffer location is always printed in the print position of the same number. Distribution of the data to accord with the format of the stationery on which it is to be printed must therefore be effected by the output instructions which transfer the data from the HSS to the buffer. Examples of how this is done are shown in Appendices 7(i) and 7(ii).

CONDITION PLUGBOARD

66. The Condition Plugboard (Appendix 9) is mounted in the Central Processor behind the Card Reader/Punch. As mentioned in para 59 it transmits reading impulses arising from comparisons between reading impulses from RS1 and RS2 to the Condition buffer. In this way types of card, changes of group etc. can be detected. When the plugboard is used

to detect simply a particular type of card (usually designated by a particular punch in a given card column) reading impulses from RS2 only are used to switch the programme to a sequence particular to the type of data or card before this data is actually loaded into Group 0 areas.

67. Applications of the Condition Plugboard are given in the following examples which also illustrate the functions of the sockets which appear on the plugboard.

EXAMPLE 1 - Detecting card types at RS2

68. The card read at RS2 can be one of three types and requires a distinct group of input instructions to be used when its data is to be loaded into the HSS. The input instructions distribute the data from the Input buffer among a number of areas and are followed by processing instructions appropriate for the type of card.

69. The three types of card could be:

(1) Stock Issues - punched 0 in column 1.

(2) Stock Receipts - punched 1 in column 1.

(3) Stock Balance - punched 2 in column 1.

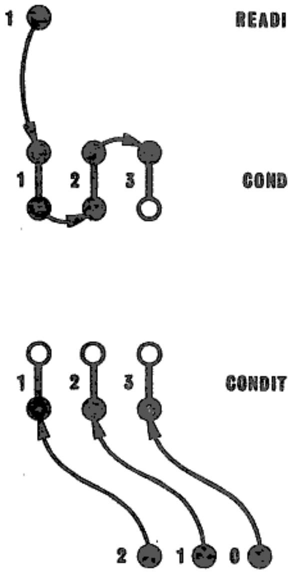
The type of card at RS2 is therefore detected by readings in column 1. These readings are then used to SET conditions for example:

(4) Stock Issues - punched 0 in column 1 - condition 03 SET.

(5) Stock Receipts - punched 1 in column 1 - condition 02 SET.

(6) Stock Balance - punched 2 in column 1 - condition 01 SET.

The wiring required on the plugboard is shown in Figure 12.



READING STATION 2 - Emits all reading impulses from column 1 of the card as it passes RS2.

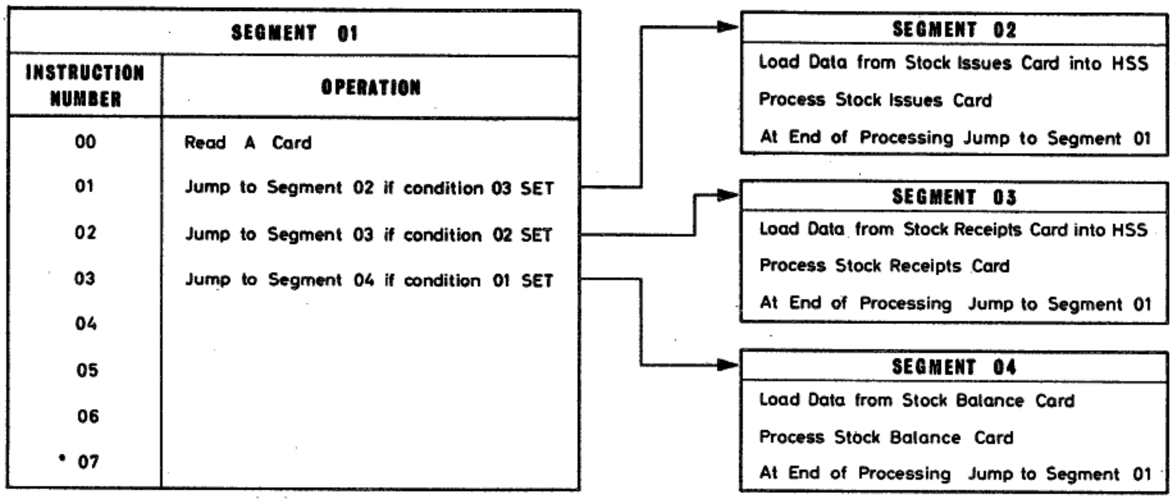
CONDITION ENTRIES - Receive reading impulses from RS2. Sockets 1, 2 and 3 receive all reading impulses from RS2 socket 1.

CONDITION CONTROLS - Filter the impulses received at the ENTRY sockets.

FE IMPULSES - Filter impulses correspond to reading impulses. When 0 FE impulse received by CONTROL co-incident with 0 reading impulse at ENTRY socket, a 1-bit loaded in to the Condition buffer.

Fig.12: Detecting card types at RS2

70. The programme for this example could be written as follows:



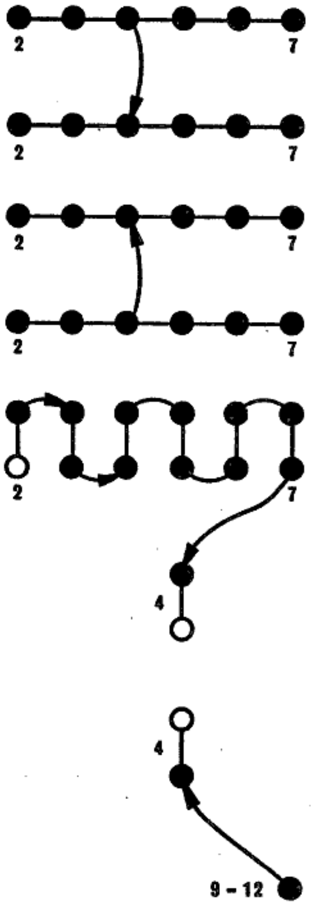
SEGMENT 01	
INSTRUCTION NUMBER	OPERATION
00	Read A Card
01	Jump to Segment 02 if condition 03 SET
02	Jump to Segment 03 if condition 02 SET
03	Jump to Segment 04 if condition 01 SET
04	
05	
06	
• 07	

SEGMENT 02
Load Data from Stock Issues Card into HSS
Process Stock Issues Card
At End of Processing Jump to Segment 01

SEGMENT 03
Load Data from Stock Receipts Card into HSS
Process Stock Receipts Card
At End of Processing Jump to Segment 01

SEGMENT 04
Load Data from Stock Balance Card
Process Stock Balance Card
At End of Processing Jump to Segment 01

EXAMPLE 2 - Detecting changes of card group



READING STATION 1 – Sockets 2 to 7 emit all reading impulses from card columns 2 to 7 as a card passes RS1.

INEQUALITY DETECTOR ENTRY 1 – Receives, on sockets 2 to 7, reading impulses from RS1.

INEQUALITY DETECTOR ENTRY 2 – Receives, on sockets 2 to 7, reading impulses from RS2.

READING STATION 2 – Sockets 2 to 7 emit all reading impulses from card columns 2 to 7 as a card passes RS2.

INEQUALITY DETECTOR EXITS – EXIT 7 emits whenever an impulse at ENTRY 1 is not coincident with an impulse at ENTRY 2.

CONDITION ENTRY – ENTRY 4 receives the impulse from the Inequality Detector Exit.

CONDITION CONTROL – Filters the impulse received at its corresponding ENTRY socket.

SYSTEMATIC COMMAND – Emits continuously from reading impulse 9 to reading impulse 12. If Inequality Detector EXIT 7 emits between these times, a 1-bit is loaded in Condition buffer location 4.

Fig: 13: Detecting changes of card group

71. In Example 1 it was assumed that three types of card were used and these could be read in random order. In this example, the cards are pre-sorted into groups, each group being identified by punches in a given card field, and sorted into the order Balance, Receipts, Issues.

72. Taking the cards used in the first example, each could be punched with a six-digit part number in columns 2 to 7 on which the cards would be sorted into groups. Without this number, all that could be obtained in Example 1 would be a list of issues and receipts with totals for each and a final new stock balance. With the part number, all of the above can be obtained with details of issues and receipts and a new stock balance for each part. The wiring required to compare part numbers and thus detect changes of group (part number) is shown in Fig. 13. This wiring is additional to that shown in Fig. 12 if all of the above information is required.

73. The programme written for Example 1 would be modified and could be written as follows:

SEGMENT 01	
INSTRUCTION NUMBER	**OPERATION**
00	Read A Card
01	Jump to Segment 05 if condition 01 NONSET (Error)
02	Jump to Segment 08 if condition 4 SET (Error)
	Load Data from Balance Card
	Process Balance Card
	Jump to Segment 02

SEGMENT 02	
INSTRUCTION NUMBER	**OPERATION**
00	Read A Card
01	Jump to Segment 06 if condition 02 NONSET (Error)
02	Jump to Segment 09 if condition 4 SET (Error)
	Load Data from Receipts Card
	Process Receipts Card
	Jump to Segment 03

SEGMENT 03	
INSTRUCTION NUMBER	**OPERATION**
00	Read A Card
01	Jump to Segment 10 if condition 03 SET (Error)
02	Jump to Segment 07 if condition 4 NONSET (Error)
	Load Data from Issues Card
	Process Issues Card
	Process for End of Group
	Jump to Segment 01 if condition 5 NONSET

Note: It is assumed that every group consists of one card of each type. Segments 05 to 10 are error routines to be followed if a card is not present or if the cards are out of order.

Condition 01 SET - Balance Card
02 SET - Receipts Card
03 SET - Issues Card
4 SET - End of Group (Fig. 12)
5 SET - East Card in File (Fig.14)

EXAMPLE 3 - Detecting the last card in a file

74. The Last Card (LC) socket emits an impulse when the last card in a file has been fed from the hopper on the main track. The use of this impulse to create a SET condition in a marker is shown in the programme of Example 2. The wiring is illustrated in Figure 14.

WIRING RESTRICTION

75. Reading impulses must never be fed to the CONDITION CONTROL sockets.

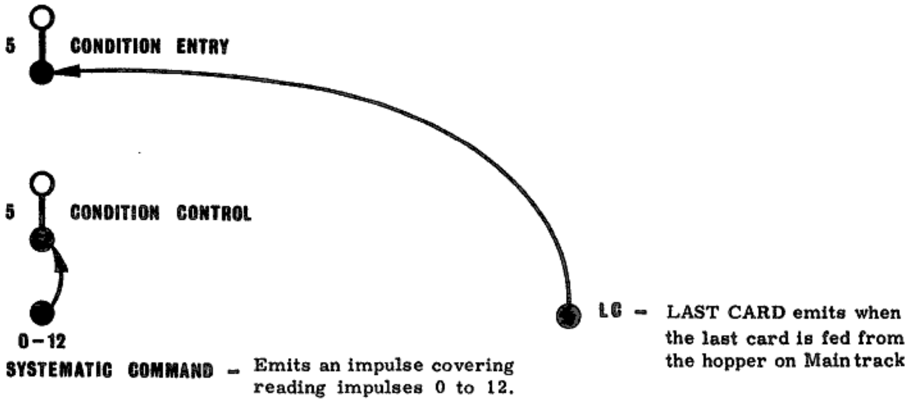

5
CONDITION ENTRY
5
CONDITION CONTROL
0-12
SYSTEMATIC COMMAND – Emits an impulse covering reading impulses 0 to 12.
LC – LAST CARD emits when the last card is fed from the hopper on Main track

Fig.14: Detecting the last card of a file

CHAPTER THREE

PROGRAMMING THE GAMMA 10

GENERAL

1. The Gamma 10 is a stored programme computer i.e. all the instructions which control the various units of the machine during a job are stored in the High Speed Store (HSS). An instruction is an order to the computer to carry out a basic operation. For example:

(1) Transfer data from one part to another part of the HSS.

(2) Add together two numbers stored in the HSS.

(3) Print a line with the data held in the Print buffer.

2. The programme is executed sequentially, instruction by instruction although in most cases, the next instruction may be executed before the work ordered by the previous instruction has been completed.

3. Before an instruction can be executed it must be transferred from the HSS to the control units (i.e. staticized) this process is automatic and followed by:

(1) Analysis of the instruction.

(2) Execution of the operation which the instruction defines or transmission of the instruction to the required input/output unit.

The execution of an instruction is effected by the arithmetic and control units.

PROGRAMME INSTRUCTIONS

4. The Gamma 10 instruction is defined by three characters.

(1) O defining the type of operation.

(2) A

(3) B

Each of these characters is allocated a value between 00 and 63 to compile a set of fifty-five different instructions and operations. A code chart summarizing the codes of these instructions is given in Appendix 3.

5. In this chapter, the programme instructions are described individually and in groups according to their use. Where necessary, an example is given to show the application of an instruction. Examples of the use of groups of instructions are given in Appendices 5, 6 and 7.

6. Throughout this chapter, in the descriptions of the instructions and in the examples, the operating times are given in terms of High Speed Store cycles i.e. 7-microsecond cycles.

INPUT INSTRUCTIONS

7. These instructions are used to cause cards to be read and data to be transferred into the HSS. There are three input instructions:

(1) Read A Card (RAC) which transfers the content of the Condition buffer into markers 1 to 32 in the HSS.

(2) Input Numeric Characters (INC) which is used for transferring data from the Input buffer into a Group 0 area.

(3) Input All Characters (IAC) which transfers any type of data from the Input buffer into a Group 0 area.

For a given card, RAC is always followed by two or more input instructions (INC and, or IAC) RAC is generally the first instruction of segment 01 (the first segment).

8. When the programme has been loaded and the cards to be processed have been placed in the hopper the START button is pressed. Automatically, cards are fed and data read from the first card as it passes RS2 is loaded into the Input buffer and the Condition buffer. The first programme instruction (RAC) is then executed and this transfers the content of the Condition buffer to markers 1 to 32 in the HSS. Instructions to identify the first card and verify that it is in fact the correct card will then be executed. These instructions are followed by two or more instructions (INC and, or IAC) which transfer the data for the first card from the Input buffer to Group 0 areas in the HSS. This data is processed according to the instructions which follow the IAC and INC instructions.

9. When the INC and IAC instructions are executed, the computer counts automatically the number of Input buffer locations which they cover. When this count reaches 80 (80 locations in the Input buffer) cards are again fed so that the Input and Condition buffers are loaded with data from the second card. The process of transferring the content of the Condition buffer, identifying and verifying the card, transferring and processing the data from it is then repeated and continues until the hopper is empty.

10. Examples of the use of input instructions are given in Appendices 5(i) and 5(ii).

RAC - Read A Card

This instruction transfers the content of the Condition buffer to markers 1 to 32 and clears the way for a card to be read. During the course of a job the feed cycle (one mechanical cycle) starts when the Input buffer is completely cleared. It is the next mechanical cycle, if the last input instruction for the previous card occurs before the end of the current mechanical cycle; otherwise it is delayed for one cycle.

During the feed cycle data read at RS2 is loaded into the Input and Condition buffers. The content of the Condition buffer is transferred into markers 1 to 32 by the RAC instruction.

The content of the Input buffer is transferred into Group 0 areas in the HSS by subsequent input instructions.

Timing: 150 cycles

Mnemonic	O	A	B
RAC	32	00	

INC - Input Numeric Characters

This instruction transfers the numeric content of consecutive locations (defined in B) of the Input buffer into a Group 0 area(defined in A) thus clearing these Input buffer locations.

Zeros and overpunches are not transferred, their corresponding locations in the HSS are filled with spaces (machine code 0).

Excess locations in the HSS area are filled with spaces. The clearing effect of this instruction can be used to eliminate data not required to be processed. Alternatively INC OO XX will eliminate card field of length XX.

Timing: $11 + 2a + L$ cycles if $a \geq L$

$10 + a + 2L$ cycles if $a < L$

where: a = length of Group 0 area
L = number of Input buffer locations

Mnemonic	O	A	B
INC	28	Group 0 area No.	Number of input buffer locations to be transferred.

IAC - Input All Characters

This instruction transfers the alphanumeric content of consecutive Input buffer locations (defined in B) into a Group 0 area (defined in A).

Excess locations in the Group 0 area are filled with spaces (machine code 0).

Excess locations of the Input buffer are cleared. This effect can be used for eliminating data not required to be processed. Alternatively IAC OO XX will eliminate card field of length XX.

Marker characters which may have been generated in the Input buffer by error punches in the card are replaced by spaces.

Timing: 11 + 2a + L cycles if a $\geq$ L

10 + a + 2L cycles if a < L

where: a = length of Group 0 area

L = number of Input buffer locations

Mnemonic	O	A	B
IAC	29	Group 0 area No.	Number of input buffer locations to be transferred.

OUTPUT INSTRUCTIONS

11. These instructions are used to transfer data from Group 0 areas to the output buffers and to command the two output functions - printing and punching. There are six output instructions:

(1) Print (PRI) which defines which paper feed is to be used and by how many line-spaces (if any) the paper is to be moved. This instruction also prepares the computer for printing.

(2) Select Stacker Punch (SSP) which defines into which stacker cards (which may not necessarily be punched) are to be selected. This instruction also prepares the computer for punching.

(3) Output All Characters (OAC) which transfers all types of coded data from a Group 0 area to either the Print or the Punch buffer.

(4) Output Numeric Unsuppressed (ONU) which transfers numeric data (including all zeros) from a Group 0 area to either the Print or the Punch buffer.

(5) Output Numeric Suppressed (ONS) which transfers numeric data from a Group 0 area to either the Print or the Punch buffer. All unwanted zeros are eliminated simultaneous with the transfer.

(6) Output Single Character (OSC) which can generate one of the sixty available characters in either the Print or the Punch buffer.

12. The buffer to which output data is to be transferred is defined by the PRI and SSP instructions. If the output data is to be transferred to the Print buffer, the PRI instruction must precede the remaining output instructions. If the output data is to be transferred to the Punch buffer, the SSP instruction must precede the remaining output instructions. When a set of data is to be both printed and punched, two series of output instructions must be used - one series preceded by a PRI instruction, the other preceded by a SSP instruction.

13. The PRI and SSP instructions can be programmed anywhere in a programme segment and need not necessarily be followed immediately by the remaining output instructions.

14. The output instructions for the Print buffer must cover its full 120 locations. As the instructions are executed, the computer counts automatically the number of buffer locations which they cover. When this count reaches 120 printing can start automatically. When printing does start, a count down begins on the number of buffer locations which are not cleared. When this count reaches zero the next PRI instruction can be executed. If the next PRI instruction is encountered before the count down reaches zero, the programme is blocked until it does and the next PRI instruction can be staticized.

15. The output instructions for the Punch buffer must cover its full 80 locations or, if over-punching is required, a multiple of 80 locations. The computer counts automatically the number of locations covered by the output instructions and punching can only begin when this count has reached 80 or a multiple of 80.

16. Punching starts in the mechanical cycle following the mechanical cycle in which the output instructions were executed. This is the next mechanical cycle if the last output instruction is followed by an RAC instruction whose following input instructions are executed before the end of the cycle. This condition also applies if the last output instruction is followed by either an IBC or an EMP instruction. If this condition is not satisfied punching is delayed for one cycle.

PRI - Print (and paper advance)

The PRI instruction defines the paper feed to be used, commands the paper movement, and causes marker 59 (PF2) or 60 (PF1) to take the condition SET when the Completed Form (CF) condition is encountered on the corresponding paper feed.

The programme is blocked on the PRI instruction if the preceding PRI instruction has not been completed. If the PRI instruction is not followed by output instructions, ONU, ONS, OAC, OSC, only the paper feed function occurs.

Paper movement occurs during the time the instructions which follow PRI are being carried out.

Example:

```
PRI 42 03  ________________________________  SKIP STARTS
                                                 |
Output instruction 1                             |
Output instruction 2                             |
Output instruction 3                             |
Output instruction 4                             |
Output instruction 5  Print buffer loaded        |
TRF 05 09                                        V
ADD 23 09  ________________________________  SKIP ENDS
                                             PRINTING STARTS
```

Timing: 15 cycles

Mnemonic	O	A	B
		PF1	PF2
PRI	34	See table below	See table below

A and B Characters	MEANING
00	No paper advance
01	1 line-space
02	2 line-spaces
03	3 line-spaces
40	The end of skip is controlled by track 0 of the paper tape loop
41	The end of skip is controlled by track 1
42	The end of skip is controlled by track 2
43	The end of skip is controlled by track 3
44	The end of skip is controlled by track 4
45	The end of skip is controlled by track 5
46	The end of skip is controlled by track 6

SSP - Select Stacker Punch

This instruction commands stacker selection and prepares the system for punching.

The content of the Punch buffer is punched in the card during the next track movement.

The programme blocks on the next SSP instruction if the preceding punching operation has not been completed.

Timing: 166 cycles.

Mnemonic	O	A	B
SSP	33		Normal Stacker - 00 Stacker - 01 Stacker - 02

OAC - Output All Characters

This instruction transfers, unaltered, all coded data in a Group 0 area (defined in A) to a number of consecutive locations of either the Print or Punch buffer.

The instruction operates from right to left in the HSS first loading Punch buffer location 80 or Print buffer location 120. The first character transferred by the next output instruction is loaded into the output buffer immediately to the left of the last location counted.

Instruction OAC OO XX has no effect i.e. no characters are transferred but the associated count progresses by length XX.

Excess locations in the output buffer remain blank but are counted. This effect can be used for skipping output buffer areas which are not used.

Timing: 9 + 2L cycles

where L = number of output buffer locations

Mnemonic	O	A	B
OAC	24	Group 0 area No.	Length of output buffer area

ONU - Output Numeric Unsuppressed

This instruction transfers the numeric content of a Group 0 area (defined in A) to a number of consecutive locations of either the Print or Punch buffer.

When the instruction is carried out, all blank locations in the Group 0 area are initially filled with coded zeros (machine code 16). The area content is then transferred to the output buffer. At the end of the instruction, all coded zeros in the area are cleared. Therefore, a card index which contains coded zeros and which was loaded into the HSS by an IAC instruction is modified by an ONU instruction.

Instruction ONU OO XX has no effect; i.e. no characters are transferred, but the associated count progresses by length XX.

Excess locations in the output buffer remain blank but are counted. This effect can be used for skipping output buffer locations which are not used.

Timing: $14 + 5a + L$ cycles if $a > L$

$14 + 4a + 2L$ cycles if $a \leq L$

where: a = length of Group 0 area
L = number of output buffer locations

Mnemonic	O	A	B
ONU	27	Group 0 area No.	Length of output buffer area

ONU

ONS - Output Numeric Suppressed

This instruction transfers the numeric content of a Group 0 area (defined in A) to a number of consecutive locations in either the Print or Punch buffer.

Zeros and spaces to the right of the most significant digit are coded as zeros.

Any non-numeric character to the right of the most significant digit is transferred. In any other position it is replaced by a space.

All coded zeros in the Group 0 area are eliminated.

Timing: 14 + 5a + L cycles if $a > L$

14 + 4a + 2L cycles if $a \leq L$

where: a = length of Group 0 area.
L = number of output buffer locations

Mnemonic	O	A	B
ONS	25	Group 0 area No.	Length of output buffer area

OSC - Output Single Character

This instruction can generate any of the sixty possible characters either, in a blank location in the Print or Punch buffer or, an 11 and/or 12 overpunch in a location of the Punch buffer already holding numeric data.

Overpunches generated in the Punch buffer do not destroy the previous content of the locations in which they are generated.

The character generated is stored in the location immediately to the left of the last location counted in the previous output instruction.

Up to 63 output buffer locations can be skipped using an OSC with A = 00.

The machine codes (27, 31, 59, 63) corresponding to the four possible states of markers can be specified in A of the OSC instruction, provided it is intended for the Punch buffer. When any one of these values is specified, both 11 and 12 overpunches are generated in the same location of the Punch buffer.

WARNING

> An attempt to generate a marker code in a Print buffer location causes an error halt.

Timing: 7 + 2L cycles

where L = number of buffer locations to be counted.

Mnemonic	O	A	B
OSC	26	Machine code of character	Number of buffer locations to be counted

TRANSFER INSTRUCTIONS

17. These instructions are used to copy the content of one area into another area without destroying the content of the source area.

18. In every transfer instruction the source area is defined in A and the receiving area is defined in B. The content of the source area specified in A is therefore copied in the receiving area specified in B.

19. Data may be copied directly from one Group 0 area into another Group 0 area or into an area in Group 1, Group 2 or Group 3 and vice versa. Data can never be copied directly from an area in Group 1, Group 2 or Group 3 into another area in one of these groups. Copying operations must therefore always involve a Group 0 area.

TRF - Transfer

This instruction copies the content and sign of one Group 0 area (defined in A) into another Group 0 area (defined in B). The content of the area defined in A is not destroyed.

A Group 0 area can be cleared and its sign made positive using a TRF instruction with A = 00.

The instruction operates from right to left, i.e. the least significant character is transferred first.

Excess locations in the receiving area are filled with spaces if the reveiving area is longer than the source area.

Excess locations in the source area are not transferred if the source area is longer than the receiving area.

Example TRF 24 49

	Area 24						Area 49								
Before	L	O	N	D	O	N	1	2	C	H	I	C	A	G	O
After	L	O	N	D	O	N				L	O	N	D	O	N

Timing: 15 + 2b + a cycles if a < b

15 + 3b cycles if a ≥ b

where: a = length of source area
b = length of receiving area

Mnemonic	O	A	B
TRF	08	Group 0 source area No.	Group 0 receiving area No.

TF1 (2) (3) - Transfer From Group 1 (2) (3)

These instructions copy the content and sign of a Group 1, Group 2 or Group 3 area (defined in A) into a Group 0 area (defined in B).

Timing: 15 + 2b + a cycles if a < b

15 + 3b cycles if a ≥ b

where: a = length of source area
b = length of receiving area

Mnemonic	O	A	B
TF1	09	Group 1 area No.	Group 0 area No.
TF2	10	Group 2 area No.	Group 0 area No.
TF3	11	Group 3 area No.	Group 0 area No.

MARKER INSTRUCTIONS

20. These instructions allow signs and conditions to be transferred from marker to marker and signs and conditions to be changed according to other signs and conditions. They can only be used to affect markers 01 to 63. Marker 00 is always SET and POSITIVE.

21. The signs of markers in Group 1, Group 2 and Group 3 can not be altered by these instructions.

22. In these instructions, A defines the source or reference marker which is not altered, B defines the receiving marker which is to be altered by the instruction.

TCC - Transfer Condition to Condition

This instruction copies the condition in a source marker (defined in A) into a receiving marker (defined in B). The condition in the source marker is not altered.

The condition in marker 00 is always SET.

Any condition in a marker can be SET using a TCC instruction with A = 00

Example

TCC 10 16

Before

Marker	10	16
Condition	NONSET	SET

After

Marker	10	16
Condition	NONSET	NONSET

Timing: 12 cycles

Mnemonic	O	A	B
TCC	36	Marker No.	Marker No.

ASC - Allocate Sign according to Condition

This instruction allocates the sign in a receiving marker (defined in B) in accordance with the condition in a source marker. The source marker is not altered.

If the condition in the source marker is SET the sign in the receiving marker becomes negative.

If the condition in the source marker is NONSET the sign in the receiving marker becomes positive.

Any sign in a marker can be made negative using a ASC instruction with A = 00.

Example:

ASC 42 10

Before

Marker	10	42
Condition	NONSET	NONSET
Sign	-	+

After

Marker	10	42
Condition	NONSET	NONSET
Sign	+	+

Timing: 12 cycles

Mnemonic	O	A	B
ASC	37	Marker No.	Marker No.

ACS - Allocate Condition according to Sign

This instruction allocates the condition in a receiving marker (defined in B) in accordance with the sign in a source marker. The sign in the source marker is not altered.

If the source marker is positive, the condition in the receiving marker is NONSET.

If the source marker is negative, the condition in the receiving marker is SET.

Any condition in a marker can be NONSET using a ACS instruction with A = 00.

Example:

ACS 46 16

Before

Marker	46	16
Condition	NONSET	SET
Sign	+	+

After

Marker	46	16
Condition	NONSET	NONSET
Sign	+	+

Timing: 12 cycles

Mnemonic	O	A	B
ACS	38	Marker No.	Marker No.

TSS - Transfer Sign to Sign

This instruction copies the sign in a source marker (defined in A) into a receiving marker (defined in B). The sign in the source marker is not altered.

Marker 00 is SET + when the computer is switched on and can not be altered.

Any marker can be made positive using a TSS instruction with A = 00.

Example:

TSS 10 16

Before

Marker	10	16
Sign	-	+

After

Marker	10	16
Sign	-	-

Timing: 12 cycles

Mnemonic	O	A	B
TSS	39	Marker No.	Marker No.

ARITHMETIC INSTRUCTIONS

23. These instructions enable the computer to perform the four arithmetic operations Add, Subtract, Multiply and Divide and also to compare the numeric values and signs of two operands.

24. The operands must always be stored in Group 0 areas.

ADD - Add

This instruction performs algebraic addition of two numbers each stored in a Group 0 area.

The augend (specified in B) is replaced by the sum. The addend is not altered.

The decimal points of the two operands must line up.

The operands may be of any length provided the area occupied by augend is of sufficient length to accommodate the sum. Capacity overflow causes the computer to stop and the CAP O'FLOW lamp to glow.

A zero sum is always allotted the sign of the augend.

To double a number it is sufficient to specify in both A and B the number of the area holding the number to be doubled.

Example:

ADD 54 59

	Area 54					Area 59						
Before	1	2	9	8	+		7	5	3	2	4	+
After	1	2	9	8	+		7	6	6	2	2	+

Timing: 19 + 5b + a cycles if a ≤ b with complementing
19 + 6b cycles if a > b with complementing
13 + 2b + a cycles if a ≤ b with no complementing
13 + 3b cycles if a > b with no complementing

where: a = number of digits in Addend
b = number of digits in Augend

Mnemonic	O	A	B
ADD	16	Addend Area No.	Augend Area No.

SUB - Subtract

This instruction performs algebraic subtraction of two numbers each stored in a Group 0 area.

The minuend (specified in B) is replaced by the result. The subtrahend is not altered.

The decimal points of the two operands must line up.

The operands may be of any length provided the area occupied by the minuend is of sufficient length to accommodate the difference. Capacity overflow causes the computer to stop and the CAP O'FLOW lamp to glow.

A zero difference is always allotted the sign of the minuend.

If, in the instruction, the same area number is allotted to both A and B this area is cleared but retains the sign of the initial operand.

Example:

SUB 09 50

	Area 09						Area 50							
Before	0	2	7	1	4	-			7	3	2	1	4	-
After	0	2	7	1	4	-			7	0	5	0	0	-

Timing: 13 + 2b + a cycles if a ≤ b with no complementing
19 + 5b + a cycles if a ≤ with complementing
13 + 3b cycles if a > b with no complementing
19 + 6b cycles if a > b with complementing

Mnemonic	O	A	B
SUB	17	Subtrahend Area No.	Minuend Area No.

MUL - Multiply

This instruction performs algebraic multiplication of two operands each stored in a Group 0 area.

The operands are not destroyed by the operation.

The product is formed in area 63 which is cleared automatically at the start of the operation. Area 63 must have a length of at least:

length of multiplicand area + number of digits in multiplier + 1

If the actual length of the multiplier is not known the length of the area which it occupies must be used in time calculations. Capacity overflow stops the computer and is signalled on the control console.

Area 63 can be used to store the multiplier which is then destroyed by the operation.

A number can be squared by specifying the same area in both A and B of the instruction. This does not apply when the number is stored in area 63.

The number of decimals in the product is:

the sum of the number of decimals in the multiplier and multiplicand.

Example:

MUL 24 20

Before

Area 20

		1	2	+

Area 24

4	8	1	2	9	-

Area 63 assuming length of multiplier not known: $a + b + 1 = 5 + 4 + 1 = 10$

After

Area 20

		1	2	+

Area 24

4	8	1	2	9	-

Area 63 (Product)

				5	7	7	5	4	8	-

Timing: $35 + 5.L63 + a + 2$ (Real length of multiplier) $+ \Sigma$ Mr digits $(3a + 10)$ if $a \leq b$
$35 + 5.L63 + b + 2$ (Real length of multiplier) $+ \Sigma$ Mr digits $(3a + 10)$ if $a > b$

where: L63 = the length of Group 0 area 63

Mnemonic	O	A	B
MUL	20	Multiplicand Area No.	Multiplier Area No.

DIV - Divide

This instruction performs algebraic division of two operands each stored in a Group 0 area.

The dividend is replaced by the remainder (set to the right).

The dividend must be stored in area 63 prior to starting the operation. It must also be correctly positioned in area 63 to ensure a quotient to the required degree of accuracy (number of decimals). The number of decimals which will appear in the quotient is given by the formula:

No. of quotient decimals = No. of dividend decimals - No. of divisor decimals.

If, for example, we have a 3 decimal dividend and a 2 decimal divisor and require a 2 decimal quotient the dividend will have to be shifted to the left one location. This is achieved using instruction SHL 63 0 1 (page 68).

The area to be occupied by the quotient is automatically cleared at the start of the operation. This area can be the divisor area, in which case the divisor is destroyed, or area 63 in which case the remainder is destroyed.

The CAP O'FLOW lamp on the control console glows to indicate capacity overflow in the quotient area.

Both the number of decimals required and the number of integers which will appear must be taken into account when calculating the length of the quotient area. The number of integers is calculated using the formula:

Number of quotient integers $I_Q = I_D - I_d + 1$

where: I_d = maximum No. of divisor integers.
I_D = No. of dividend integers.

For example, with a 6 integers and 3 decimals divisor and a 1 integer 2 decimals dividend the quotient can have a maximum of 6 - 1 + 1 = 6 integers. If we require 2 decimals in the quotient then the length of its area must be 6 + 2 = 8 locations.

Example:

Initial Conditions

Area 37 (6 locations) - DIVIDEND											3	5	2	4	1	+
Area 32 (4 locations) - DIVISOR												1	3	5	2	-
Area 40 (9 locations) - QUOTIENT								A	B	C	D	E	F	G	H	+
Area 63 (15 locations)	0	0	0	0	7	5	0	1	2	6	8	5	1	4	9	+

Instructions:

TRF 37 63

SHL 63 02

DIV 32 40

Final Conditions

Area 37 (6 locations) - DIVIDEND											3	5	2	4	1	+
Area 32 (4 locations) - DIVISOR												1	3	5	2	-
Area 40 (9 locations) - QUOTIENT							0	0	0	0	0	2	6	0	6	-
Area 63 (15 locations) - REMAINDER	0	0	0	0	0	0	0	0	0	0	0	0	7	8	8	+

Timing: $34 + 20.L63 + 2b + a\,(6.L63 - 6a - 16) + \Sigma\, Q(3a + 9)$ cycles

where: L63 = the length of Group 0 area 63
a = the number of Divisor digits
b = the number of Dividend digits

Mnemonic	O	A	B
DIV	21	Divisor Area No.	Quotient Area No.

CMP - Compare

This instruction compares the content and signs of two Group 0 areas. The result, which is stored until the next comparison occurs, indicates that the operand specified in A is = ≠ > < ≥ ≤ to the operand specified in B.

The operands are not altered by the operation.

Comparison between two zero operands gives the result = irrespective of the signs of the operand.

The areas can be of any length. If they differ in length, excess locations are compared with the space characters.

Instruction CMP A OO compares the operand specified in A with zero. The result > indicates a positive quantity in A.

Comparison of alphanumeric data can give the result = or ≠

Timing: 13 + a + 2b cycles if the operands have the same sign
13 + a + b cycles if operands have different signs

where: a = No. of locations in Area A
b = No. of locations in area B

Mnemonic	O	A	B
CMP	19	Area No. (first operand)	Area No. (second operand)

JUMP INSTRUCTIONS

25. There are two types of jump instruction:

(1) Instruction jumps which are used to jump the programme within a segment either forward or back a number of instructions.

(2) Segment jumps which are used to jump the programme from segment to segment and whose principal function is to bring subroutines into operation. Segment jump instructions always jump the programme to the first instruction (00) of the destination segment. Every segment, except the last, must have a segment jump instruction as its last instruction to jump the programme to the next segment.

26. Both types of jump instruction can be either unconditional i.e. whenever the instruction is encountered a jump always takes place, or conditional according to the result of a comparison or the sign or condition within a marker.

JI = (≠), (>), (≥), (<), (≤) - Jump Instruction

These jump instructions use the result of the last Comparison (CMP) operation to cause a jump to another instruction (specified in B) within the same segment.

If the required condition is not present the next instruction in sequence is staticized.

The destination instruction can appear before or after the jump instruction.

A jump instruction must not jump to itself.

Example:

Instruction 24 (JI 04 16) causes a jump back to instruction 16 if the result of the comparison CMP 12 23 gave the result >

```
    SEGMENT  4
INSTRUCTION  00
             01
             :
             15
             16
             17    CMP   12   23
             18
             :
             24    JI>   04   16
             25
             :
             38
```

Timing: 15 cycles if the jump occurs
5 cycles to continue in sequence

There are six different types of instruction jump after comparison. These instructions have the same O character but different A characters corresponding to the six possible uses of the comparison result.

Mnemonic	O	A	B
JI <	00	01	Destination Instruction (00-63)
JI =	00	02	
JI ≤	00	03	
JI >	00	04	
JI ≠	00	05	
JI ≥	00	06	

JIU -Jump Instruction Unconditional

This instruction causes an automatic jump, within the same segment to the instruction specified by B.

Timing: 15 cycles

Mnemonic	O	A	B
JIU	00	07	Destination Instruction (00-63)

JID - Jump Instruction Dummy

This instruction has no effect unless it is changed into any other type of instruction jump by a Modify (MOD) instruction.

The A character of JID is altered by MOD which can transfer to it the content of a Group 0 area.

Example:

The JID instruction is altered by the MOD instruction when area 31 contains some data. If area 31 contains 07, instruction 16 becomes a JIU instruction and the programme then jumps to instruction 46

```
    SEGMENT  10
INSTRUCTION  00
             01
              :
             14
             15   MOD   31   01
             16   JID   00   46
             17
             18
              :
             45
             46
```

Timing: 5 cycles

Mnemonic	O	A	B
JID	00	00	Destination Instruction (00-63)

JIS - Jump Instruction SET
JIN - Jump Instruction NONSET

These instructions make use of the condition within a marker (i.e. SET or NONSET) to cause a jump to an instruction (specified in B) in the same segment. If the required condition is not present, the next instruction in sequence is staticized.

JIS causes a jump if the required condition is SET.

JIN causes a jump if the required condition is NONSET

Timing: 18 cycles
8 cycles continuing in sequence

Mnemonic	O	A	B
JIS	06	Marker to be tested (01-63)	Destination Instruction (00-63)
JIN	04	Marker to be tested (01-63)	

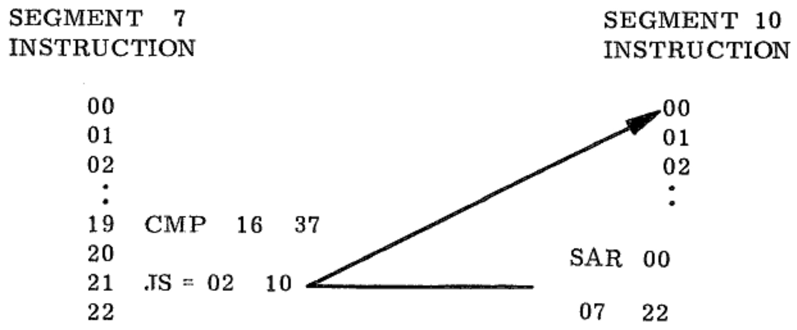
JS = (≠), (>), (≥), (<), (≤) - Jump Segment

These jump instructions use the result of the last comparison (CMP) operation to cause a jump to the first instruction of another segment (specified in B).

If the required condition is not present the next instruction in sequence is staticized.

If the jump occurs the address of the next instruction in the original segment is stored automatically in SAR 00.

Example:



Instruction 21 (JS = 02 10) of segment 7 causes a jump to the 1st instruction of segment 10 if the result of the last comparison instruction (i.e. instruction 19) before instruction 21 gives A = B. If the jump occurs, the address of instruction 22 of segment 7 is stored in SAR 00. If the result of the comparison is other than A = B instruction 22 of segment 7 is staticized.

Timing: 19 cycles
5 cycles continuing in sequence

There are six different type of segment jumps. They have the same O character but different A characters corresponding to the six possible uses of the comparison result.

Mnemonic	O	A	B
JS <	01	01	Destination Segment (01-60)
JS =	01	02	
JS ≤	01	03	
JS >	01	04	
JS ≠	01	05	
JS ≥	01	06	

JSU - Jump Segment Unconditional

This instruction causes an automatic jump to the first instruction of the segment specified in B.

When the jump occurs, the address of the next instruction in the original segment is automatically stored in SAR 00.

Timing: 19 cycles

Mnemonic	O	A	B
JSU	01	07	Destination Segment (01-60)

JSD - Jump Segment Dummy

This instruction has no effect unless it is changed into any other type of segment jump by a Modify (MOD) instruction.

The A character of JSD is altered by MOD which can transfer to it the content of a Group 0 area.

Example:

```
                 SEGMENT  10
             INSTRUCTION  00
                          01
                          .
                          .
                          17
                          18
                          19
                          .
                          .
                          31
                          32   MOD   28   01
                          33   JSD   01   12 becomes
                                 JSU   07   12
```

The MOD instruction inserts the content of area 28 in the A character of the JSD instruction. If area 28 contains 07 instruction 33 becomes a JSU instruction and the programme then jumps to the first instruction of segment 12.

Timing: 5 cycles

Mnemonic	O	A	B
JSD	01	00	Destination Segment (01-60)

JSS - Jump Segment SET
JSN - Jump Segment NONSET

These instructions make use of the condition within a marker (i.e. SET or NONSET) to cause a jump to the first instruction of another segment (specified in B).

If the jump occurs, the HSS address of the next instruction in the original segment is automatically stored in SAR 00.

JSS causes a jump if the required condition is SET.

JSN causes a jump if the required condition is NONSET.

Timing: 22 cycles
8 cycles continuing in sequence

Mnemonic	O	A	B
JSS	07	Marker to be tested (01-63)	Destination Segment (01-60)
JSN	05	Marker to be tested (01-63)	

SUBROUTINE RETURN INSTRUCTIONS

27. These instructions allow a number of segment re-entry points to be stored and enable the programme to jump from segment to segment without of necessity entering the destination segment at instruction 00.

SRA - Store Return Address

This instruction transfers the content of one SAR (specified in A) to another SAR (Specified in B).

This instruction is used if the programme includes several segments, which are accessed by Jump Segment (JS) instructions. In these cases the SARs are used to store the return addresses to be used in Jump Back (JBK) instructions which allow the programme to jump from segment to segment without resuming a segment at instruction 00.

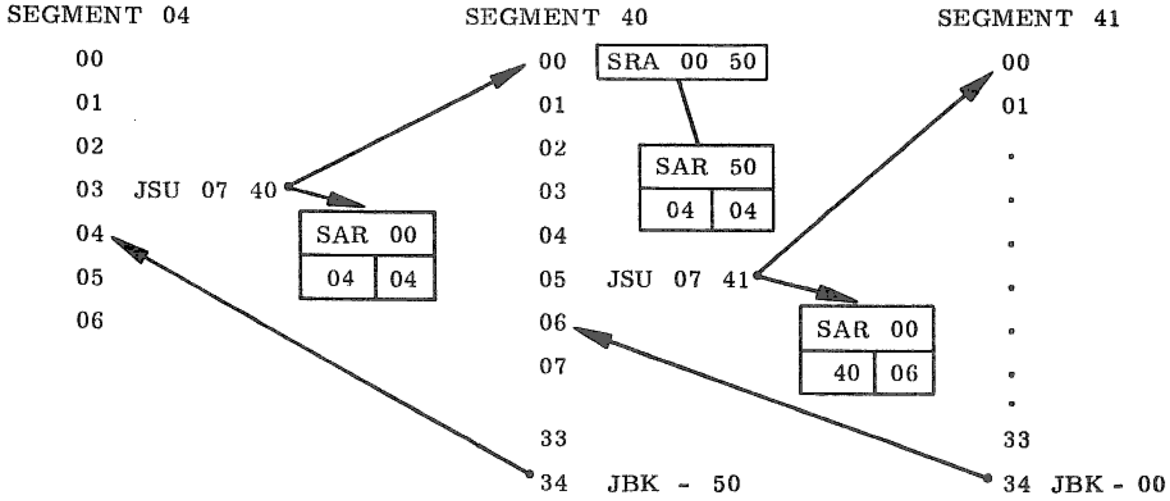
WARNING

> Neither an SAR which is in use nor SAR 61 nor SAR 62 nor SAR 63 must ever be specified in either A or B of the SRA instruction.

Example:

Segment 04 calls in segment 40 which in turn calls in segment 41. This last segment does not contain any Jump Segment instructions.

The return address to segment 40 is stored in SAR 00, the content of which is not altered during the operation of segment 41. However, the return address to segment 04 has to be transferred to another SAR (in the example SAR 50), since the content of SAR 00 will be altered during the operation of segment 40 by the JSU instruction (i.e. instruction 05) which causes a jump to segment 41.



Timing: 9 cycles

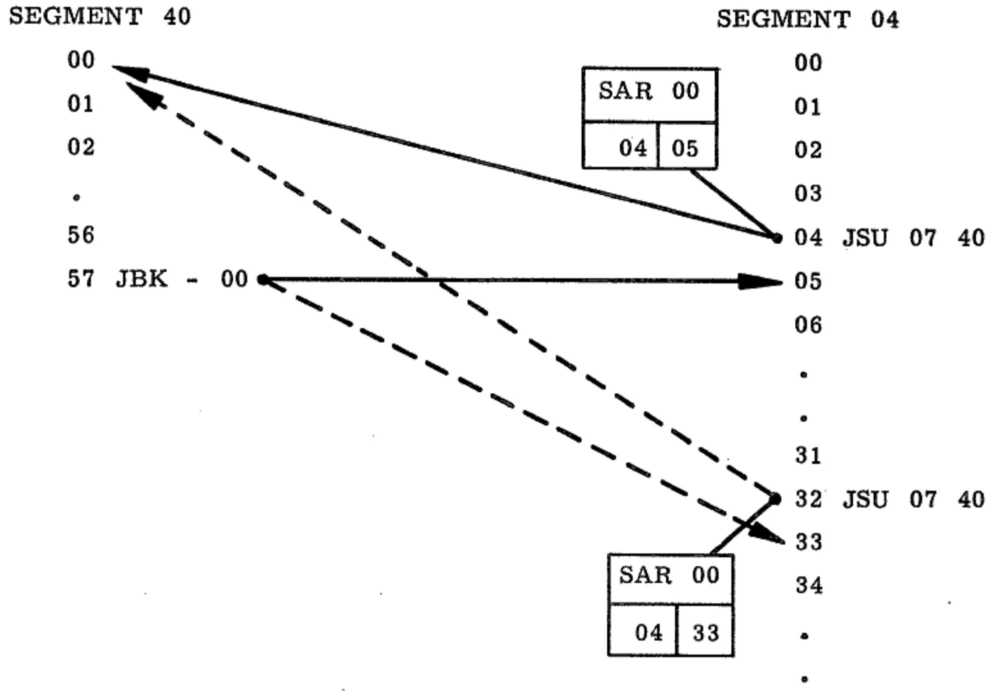
Mnemonic	O	A	B
SRA	30	Source SAR No.	Receiving SAR No.

JBK - Jump back

This instruction causes a jump to an instruction whose address was stored in SAR 00 during a segment jump. This address may have been stored in another SAR using a SRA instruction (page 63).

Example:

Segment 04 calls in segment 40 which does not contain any Jump Segment instructions. The return address to segment 04 is stored in SAR 00.



Timing: 21 cycles

Mnemonic	O	A	B
JBK	03		SAR Number

SHIFT INSTRUCTIONS

28. These instructions allow data to be shifted within Group 0 areas and additional characters to be added to data stored in these areas.

29. Provided there are sufficient excess locations to the left of the most significant digit numeric data can be multiplied by powers of ten by simple shift instructions.

30. These instructions can also be used to line-up decimals points in operands which are to be used in arithmetic operations.

31. Data can be edited and a single item of data stored within a group of items of data in one area can be extracted using these instructions.

TWA - Transfer Within Area

This instruction copies the content and sign of one Group 0 area (specified in A) into another Group 0 area (specified in B).

The excess locations of the receiving area are not destroyed.

Instruction TWA OO XX has no effect on the content of area XX but the sign of the area will be changed to +.

WARNING

The rightmost location of the receiving area must be blank.

Example:

Before

Area 15

						B	E	R	L	I	N

Area 26

	P	A	R	I	S

```
SHL  15  06
TWA  26  15
```

After

Area 15

B	E	R	L	I	N		P	A	R	I	S

Area 26

	P	A	R	I	S

Timing: 13 + 3a cycles if a ≤ b

15 + 3b cycles if a > b

where a = length of source area
b = length of receiver area

Mnemonic	O	A	B
TWA	12	Group 0 area No.	Group 0 area No.

SHR- Shift Right

This instruction shifts the content of a Group 0 area (defined in A) n locations (defined in B) to the right.

The number of characters destroyed during the operation is equal to the number of shifts.

The lefthand locations are filled with spaces.

A SHR instruction where the number of shifts is ⩾ the length (in characters) of the area clears this area.

Example:

SHR 24 03

Area 24

Before	0	7	2	8	1	3	4
After	-	-	-	0	7	2	8

Timing: $24 + 2a + 2n$ cycles if $a \geq n$
$24 + 4a$ cycles if $a < n$

where: a = length of Group 0 area
n = number of shifts

Mnemonic	O	A	B
SHR	22	Group 0 Area No.	Number of shifts (Maximum 63)

SHL - Shift Left

This instruction shifts the content of a Group 0 area (defined in A) n locations (defined in B) to the left.

The number of characters destroyed during the operation is equal to the number of shifts.

The righthand locations are filled with spaces.

A SHL instruction where the number of shifts is ≥ the length (in characters) of the area clears this area.

Example:

SHL 29 06

	Area 29					
Before	5	7	5	2	3	8
After	-	-	-	-	-	-

Timing: 26 + 3a + 2n cycles if a ≥ n
26 + 5a cycles if < n

where: a = length of Group 0 area
n = number of shifts

Mnemonic	O	A	B
SHL	23	Group 0 Area No.	Number of shifts (Maximum 63)

SCF - Shift Character Fill

This instruction inserts a character (defined in A) in the righthand location of a Group 0 area (defined in B).

Prior to inserting the character the content of the area is shifted automatically one location to the left.

A SCF instruction with A = 27, 31, 59 or 63 (marker codes) shifts the content by one location but does not insert the marker i.e. a space is inserted.

Example:

SCF 28 08

Area 08

Before	0	3	5	6	7
After	3	5	6	7	s

Timing: 24 + 3b cycles

where: b = length of Group 0 area.

Mnemonic	O	A	B
SCF	18	Character (machine code)	Group 0 Receiving Area No.

MOD - Modification (of an instruction)

This instruction modifies a parameter of a subsequent instruction by transferring the content of a Group 0 area to the parameter to be modified. The content of the source area is not destroyed. The number (ℓ) defined in B indicates the number of O, A and B parameters separating the MOD instruction from the parameter to be modified (counting in sequence B, A, O). The following rules must be applied:

(1) $\ell = 3n + 0$ if the B parameter has to be modified.

(2) $\ell = 3n + 1$ if the A parameter has to be modified.

(3) $\ell = 3n + 2$ if the O parameter has to be modified.

where n = number of instructions separating the MOD instruction from the instruction to be modified.

If ℓ = O, the B parameter of the instruction in sequence will be modified. The MOD instruction can only modify a subsequent instruction of the same segment. The length of the source area can be equal or greater than one location. If the length of the source area is one location, the content of this area will replace the parameter to be modified. If the length of the source area is greater than or equal to 2 locations, only the two righthand locations are taken into account. The numeric or alphanumeric content of these two locations will be grouped and transferred to the parameter to be modified.

Example:

Area 15

7	8	5	4	3

09	MOD	15	01	
10	ADD	42	25	The A parameter of instruction No. 10 is changed into 43 by the MOD Instruction.

Timing: 22 + 2 ℓ cycles

where ℓ is calculated according to the above formulae

Mnemonic	O	A	B
MOD	31	Group 0 Area No.	Number of O, A, B parameters

HLT - Halt

The programme halts when this instruction is detected - the condition is signalled on the control console (HALT lamp glowing). The next instruction in sequence is staticized by pressing the START push-button

Every programme must have an HLT as its final instruction.

Timing: 5 cycles.

Mnemonic	O	A	B
HLT	35		

IBC - Insert Blank Card

The Insert Blank Card (IBC) instruction commands the movement of the Secondary and Common sections of the track for one mechanical cycle. A blank card is inserted, and stops in CS4, while the card at CS4 advances to CS5. The instruction is carried out during the first mechanical cycle after the IBC instruction has been staticized.

If the IBC instruction is not staticized before 12,15 the track movement is delayed for one mechanical cycle. The card which passes the Punching Station is punched with the content of the Punch buffer.

The programme is blocked by the IBC instruction until the preceding track movement has been completed.

Timing: 6 cycles

Mnemonic	O	A	B
IBC	32	01	

EMP - Empty track

The Empty Track (EMP) instruction commands the movement of the Common section of the track for one mechanical cycle (the Primary and Secondary sections are stationary). The card in CS4 progresses during this movement and is automatically taken to the appropriate stacker.

The programme blocks on the EMP instruction until the preceding RAC or IBC instruction has been completed.

The track movement occurs during the next mechanical cycle, provided the EMP instruction has been staticized before the end of the cycle. If the EMP instruction is staticized after the end of the cycle the track movement will be delayed for one cycle.

Every programme must have EMP as its penultimate instruction.

Timing: 6 cycles

Mnemonic	O	A	B
EMP	32	02	

CHAPTER FOUR

CONTROL PANELS

CARD READER/PUNCH

PUSH-BUTTON SWITCHES

STOP	When this push-button is pressed the Card Reader/Punch stops at the end of the current cycle and the SYNCH lamp glows. The power is not shut off.
FEED 1 CARD	When this push-button is pressed one card is fed and stops in front of RS1. The card previously stopped in front of RS2 does not move. The light incorporated in the push-button is extinguished when the card has been fed.
CLEAR PRIMARY	This push-button must only be pressed when the machine is stopped. When it is pressed the Primary and Common sections of the track are cleared. The cards thus ejected are not read. This push-button also clears the Input buffer and extinguishes the READ CHECK and PUNCH CHECK lamps (if lit).
CLEAR SEC'Y	This push-button must only be pressed when the machine is stopped. When it is pressed the Secondary and Common sections of the track are cleared and the PUNCH CHECK lamp is extinguished (if lit).

PRINTER

PUSH-BUTTON SWITCHES

PF1 **PF2**	Paper Feed 1 and 2 (PF1 and PF2) push-button switches when pressed set the corresponding paper web automatically at the last printing line on a sheet.
STOP	When this push-button is pressed the Printer stops at the end of the current printing cycle. At the same time the SYNCH lamp glows. The power is not shut off.

CONTROL CONSOLE

CARD READER/PUNCH

Lamps

PRIMARY TRACK	This lamp glows: (1) Together with the START lamp when there is a card slot jam on the Primary section of the track or when the hopper is empty. (2) Together with the SYNCH and START lamps when there is a card jam between CS1 and CS4 on the Primary section of the track or at CS3′ on the Secondary track.
SEC'Y TRACK	This lamp glows: (1) Together with the START lamp when there is a card slot jam on the Secondary section of the track or when the hopper is empty. (2) Together with the SYNCH and START lamps when there is a card jam on the Secondary section of the track i.e. at CS1′ or CS2′.
RECEP	This lamp glows: (1) Together with the START lamp when stacker S1, S2 or the Normal pocket is full. (2) Together with the SYNCH and START lamps when there is a card jam on the common section of the track i.e. from CS5 onwards.

Illuminated push-button switches

The lamps incorporated in each of these push-button switches glows for as long as the push-button is pressed.

SELECTION SUPPRESS	When this push-button is pressed any programme commands for card selection are suppressed and all cards are ejected into the Normal pocket.
PUNCH SUPPRESS	When this push-button is pressed all programme punching commands are suppressed.
IBC SUPPRESS	Pressing this push-button suppresses all programme commands for inserting blank cards. Thus where these commands occur gaps between the cards will be created.

PRINTER

Lamps

PF1

PF2

One of these lamps glows together with START when a fault occurs on the corresponding paper feed:

(1) Tearing of the sprocket holes.

(2) Torn paper web.

(3) Incorrect positioning of the paper after a skip.

(4) End of the paper web.

RIBBON

This lamp and the lamp in the START push-button glows when the carbon ribbon either breaks or jams.

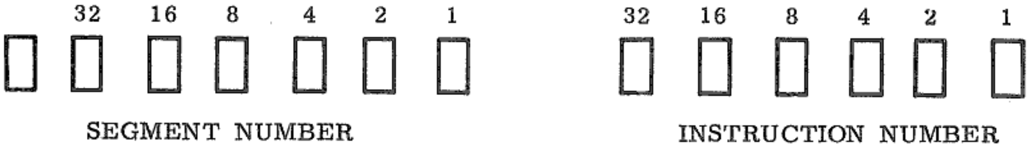
Illuminated push-button switch

PRINT SUPPRESS

Pressing this push-button suppresses printing on both paper feeds. The lamp incorporated in the push-button glows for as long as the button is pressed.

CENTRAL PROCESSOR

Lamps



The groups of lamps labelled SEGMENT NUMBER and INSTRUCTION NUMBER glow whenever the computer stops and indicate in binary the number of the segment and of the instruction which was being processed when the stop occurred.

The lamp which is not labelled is used solely for servicing and testing purposes.

LOAD ERROR

This lamp glows, together with the lamp in the START push-button, to indicate that an error has been detected during the programme control pass.

READ CHECK

This lamp glows, together with the lamp in the START push-button, to indicate that a difference has been detected between the check digits calculated at RS1 and RS2 for the same card.

PUNCH CHECK

This lamp glows, together with the START lamp to indicate that an error has been detected between the check digit calculated for a card at RS5 and the check digit for the same card calculated at RS2 or the punching station i.e. a punching error exists.

CAP. O'FLOW

This lamp glows, together with the START lamp, to indicate a capacity overflow condition during one of the four arithmetic operations.

HALT

This lamp glows, together with the START lamp, whenever a HALT instruction in the programme occurs.

This red lamp is not labelled. It glows when the POWER ON switch is pressed, and is extinguished when the computer is ready to run, i.e. when the power supplies are intact, and correct operating temperatures have been reached. During running this lamp glows whenever one of the following major failures occurs:

(1) Power drop within the machine.

(2) Abnormal temperature in the High Speed Store.

(3) Parity error in the Central Processor.

(4) Synchronization failure.

Illuminated push-button switches

MARKER 61 **MARKER 62** **MARKER 63** **LOAD MARKER**

These four push-buttons can be used to control the setting of markers 61, 62 and 63:

(1) When LOAD MARKER is pressed, the markers whose corresponding push-buttons have not been pressed are NONSET.

(2) When LOAD MARKER is pressed the markers whose corresponding push-buttons have been pressed are SET.

In this case the lamps in the MARKER push-buttons glow for as long as LOAD MARKER is pressed.

LOAD

Pressing this push-button brings into operation the internal programmes for allocating areas, loading the programme and the programme loading control programme.

2nd PASS CONTROL

Pressing this push-button brings into operation the internal programmes for controlling programme loading and area allocation.

DATA FILL

Pressing this push-button brings into operation the internal programme for loading the areas and markers.

PROCESS	Pressing this push-button: (1) Sets the Instruction Number Register to 00. (2) Sets the Segment Number Register to 01. (3) Clears all buffer stores.
SYNCH	Pressing this push-button synchronizes the drives to the Printer and Card Reader/Punch. The lamp incorporated glows whenever these drives are not synchronized.
STOP	When this push-button is pressed, the programme stops after the next RAC, IBC, EMP or PRI instruction has been staticized. When the programme stops the START lamp glows.
START	Pressing this push-button starts the programme selected by pressing LOAD, 2nd PASS CONTROL, DATA FILL or PROCESS. At the beginning of a job, pressing START causes the first card to be fed on the Primary and Secondary sections of the track. The lamp incorporated is extinguished when START is pressed. At the start of a job the lamp glows when the computer is ready to operate. During a job the lamp glows if a minor fault occurs which stops the machine.
POWER OFF	Pressing the push-button breaks the power supply to the machine. The lamp incorporated glows when the machine is powered.
POWER ON	This switch is located under the covers of the Card Reader/Punch. When it is pressed, power is supplied to the machine. The POWER OFF push-button and the red lamp glow when POWER ON is pressed.

The layout of the Control Console on the front of the Central Processor is illustrated in Figure 15 overleaf.

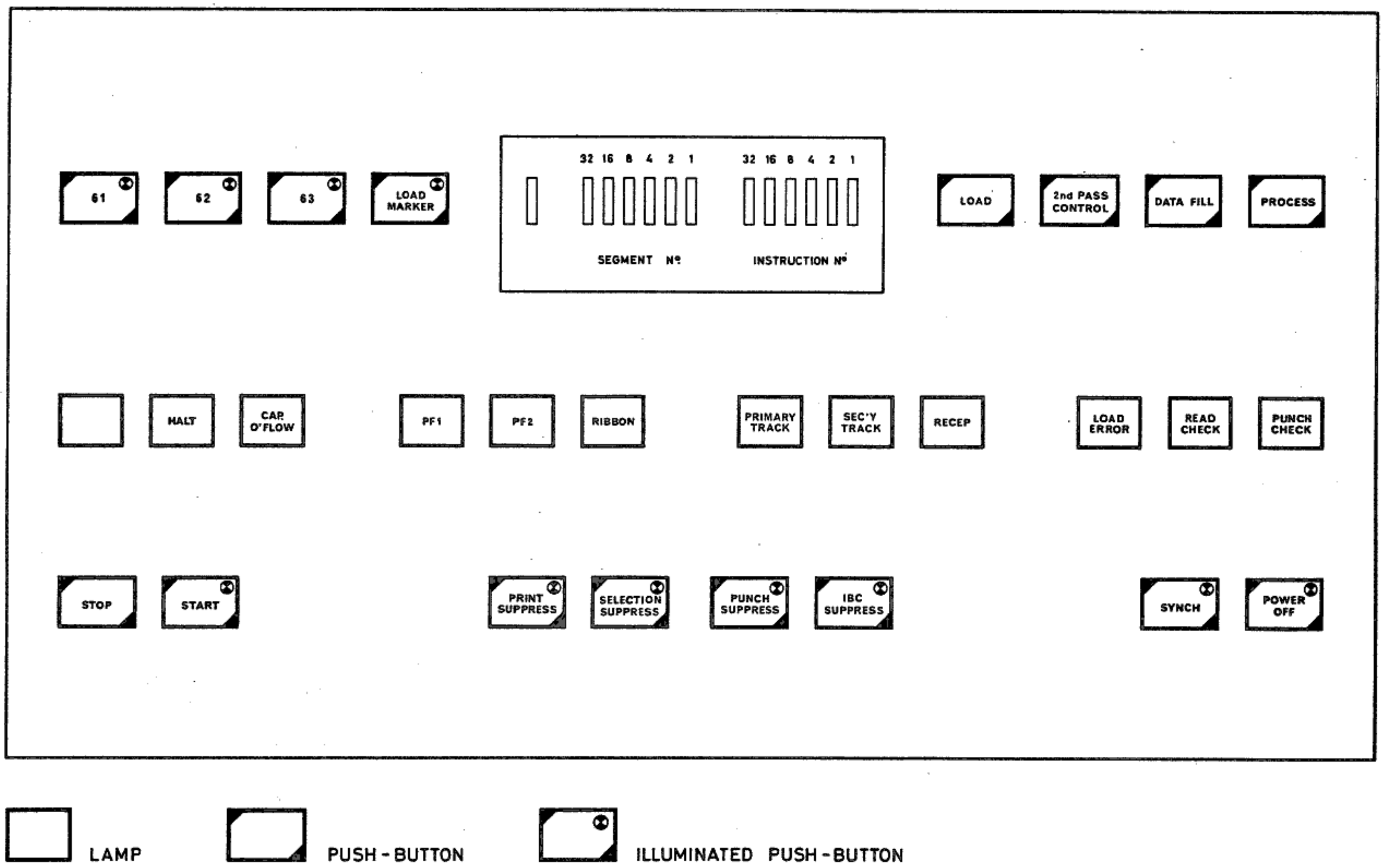


Fig.15: Control Console

CHAPTER FIVE

MANUAL OPERATIONS

SETTING-UP PROCEDURES

POSITIONING THE CONDITION PLUGBOARD

1. For technical reasons, the Condition Plugboard must be positioned when the machine is switched off. It will therefore be necessary, at the start of a job, to position the Condition Plugboard prior to pressing the POWER ON push-button. The procedure for changing the Condition Plugboard between runs is as follows:

(1) Press the POWER OFF push-button.

(2) Remove the existing Condition Plugboard.

(3) Position the new Condition Plugboard.

(4) Press POWER ON.

Data present in the HSS will not be destroyed by these operations provided this procedure is adhered to.

SWITCHING ON

2. The machine will be powered by pressing the POWER ON push-button (green push-button under the covers of the Card Reader/Punch). The non-labelled red lamp glows when this push-button is pressed. When the power supplies to the whole machine are complete, lamps incorporated in the POWER OFF and SYNCH push-buttons glow. The drives are synchronized by pressing the SYNCH push-button. The lamp incorporated in the SYNCH push-button will be extinguished when synchronization is obtained and is signalled by the lamp incorporated in the START push-button flashing. The flashing will stop and the lamp will glow continuously when the Gamma 10 is ready to operate (power supplies completed, correct temperature). The warm up period will last approximately 10 minutes when the machine is first switched on.

PRINTER

3. Prior to starting a job, the paper control tape(s) and the continuous stationery have to be positioned on the Printer.

Paper Control Tape

4. Each paper feed is controlled by a paper control tape. This is a standard 8-track paper tape (tracks numbered 0 to 7). Some of the tracks are allocated to a standard function. For example, track 6 is allocated to the last printing line on a sheet and track 7 is used for the Completed Form (CF) function.

5. The format of the paper control tape must be included in the Programme Dossier and

the tape to be used must be punched in accordance with this document.

6. The length of the paper control tape must be between 60 line-spaces (minimum) and 120 line-spaces (maximum). It will therefore be necessary, if the length of the sheet is less than 60 line-spaces, to repeat the punchings relative to a sheet several times in order to obtain a paper control tape between 60 and 120 line-spaces long. For example, for a sheet having 30 line-spaces it will be necessary to repeat the punchings twice i.e. 60 line-spaces.

Note... the length of a sheet must be a multiple of 3 line-spaces.

7. The paper tape can be punched on the hand punch provided with the Gamma 10.

8. The paper tape has to be joined in a continuous loop which has to be positioned on the appropriate paper tape mechanism.

WARNING

> When punching a paper control tape it is advisable to mark the upper surface of the tape OUTSIDE to ensure that when the loop is completed this is the outer surface of the loop.

9. The paper control tape will be positioned on the punching in track 6 by pressing the push-button located at the side of the paper tape mechanism.

Continuous Stationery

10. The pack of continuous stationery has to be placed on the tray to the front of the Printer. This tray can be moved forward, thus giving access to the paper track, by means of a lever.

11. The first sheet must be passed to the back of the print drum and postioned on the appropriate pin-wheels.

12. The tray must be moved back into its correct position by depressing the pedal to the left of the paper tray.

13. The paper must now be synchronized with the paper control tape. The last printing line has a corresponding reference marking in the margin. There are 111 line-spaces between this reference marking and the real level of the last printing line on PF1 and 81 line-spaces on PF2. This reference mark must be aligned with the reference marking appearing on the tractors.

14. The coincidence of these reference marks indicates that the printing line is the last possible line. We have already seen that the paper tape will be positioned on the punching corresponding to this line. Synchronization between the paper and the paper control tape is thus obtained.

Note... the Printer can be set-up during programme loading.

CARD READER/PUNCH

15. No special manual operations are required. Make sure that there are cards in the secondary track hopper (if required).

STARTING A JOB

16. A set of four interlocking push-buttons is used for starting a job.

ALLOCATING HSS AREAS AND PROGRAMME LOADING

17. The programme card pack consists of 2 sets of cards, each set consisting of instruction and area cards. These sets are identical except for column 67 i.e. the Set 1 cards have a 12/67 and the Set 2 cards an 11/67.

18. The loading procedures are as follows:

(1) Place the programme card pack (Set 1 leading) in the hopper.

(2) Press LOAD and then START. This causes the Set 1 cards to be loaded. The internal control programme is started when the 11/67 of the first card of Set 2 is read. This control programme compares the contents of the HSS with the contents of the cards being read (i.e. Set 2 cards).

19. Several cases may arise during loading.

(1) The operation occurs without any incidents - the programme loading and the area allocations are correct.

(2) An HSS incident occurs - the service engineer must be called.

(3) A read error occurs (READ CHECK lamp glows). Loading has to be re-started from the beginning. The procedure is as follows:

(a) Remove the cards from the hopper and clear the track by pressing CLEAR PRIMARY. The internal loading and control programmes are reset to their initial conditions by pressing PROCESS and then LOAD.

(b) Replace the cards in the hopper and press START.

Note... the service engineer must be called if the same incident recurs several times in a run.

(4) An error has been detected during the control programme run. This is indicated by a machine stop and the LOAD ERROR lamp glowing. This incident can be caused by one of the following:

(a) Set 1 has been badly loaded.

(b) Sets 1 and 2 are not identical (sort error, punch error).

The cause of the incident can be found by passing the programme pack a second time after pressing 2nd PASS CONTROL (internal control programme reset to its initial state) and START. The incident must recur:

(i) It is caused by a Set 1 card (12/67). This indicates that loading is erroneous and the service engineer has to be called.

(ii) It is caused by a Set 2 card (11/67). This card and its corresponding Set 1 card are not identical (sort error, punch error). Locate erroneous card and correct it. Restart loading from the beginning.

Note... the following procedure could be used: When the machine stop occurs, check the sort order and the identity of the error card and its corresponding Set 1 card. If there is no sort error, proceed as in (4) above. If there is a sort error. correct it and restart loading from the beginning.

LOADING THE AREAS AND THE MARKERS

20. The internal programme for loading Group 0 areas and markers is brought into operation by pressing DATA FILL and START. This internal programme allows the areas and markers to be loaded. There are two types of data fill cards:

(1) Data fill (area) cards, which load an algebraic quantity (the condition of the marker is not altered but the sign may be altered).

(2) Data fill (marker) cards, which load a condition into a marker (without altering the content of the associated area).

SUPPRESSING FUNCTIONS

21. The corresponding function will be suppressed when the following push-buttons (lamps incorporated) are pressed.

(1) PUNCH SUPPRESS.

(2) IBC SUPPRESS - This can be used to create gaps between cards.

(3) SELECTION SUPPRESS.

(4) PRINT SUPPRESS.

Note... there is no push-button for skip suppression. If this is required, it will be necessary to remove the paper control tape for the corresponding paper feed. All programme commands for skips will then be replaced by a single line-space.

MARKER PUSH-BUTTONS

22. These four push-buttons can be used to control the setting of markers 61, 62, 63. The marker corresponding to the push-button is SET (push-button pressed) or NONSET (push-button released) by pressing the LOAD MARKER push-button. The lamps incorporated in these push-buttons glow when the required condition is present.

STARTING PROCESSING

23. The following manual operations have to be carried out when the Gamma 10 is ready for programme running:

(1) Place the file to be processed in the hopper.

(2) Fill the hopper of the secondary track with blank cards.

(3) Press PROCESS, this sets the Instruction Number Register to 00 and the Segment Number Register to 01. Processing is started by pressing START.

CHAPTER SIX

INCIDENTS AND RESTART PROCEDURES

1. In this chapter we deal with two types of incidents:

(1) Incidents during programme loading and testing.

(2) Incidents during processing.

Note... incidents caused by faulty push-buttons will not be dealt with since they require the intervention of a service engineer.

INCIDENTS DURING PROGRAMME LOADING AND TESTING

2. If one of the following occurs, and before calling the service engineer ensure that the manual operations have been carried out correctly.

(1) The POWER OFF lamp does not glow.

(2) The SYNCH lamp does not glow or it glows for too long a period.

(3) The warm-up period (indicated by the START lamp flashing) exceeds 10 minutes.

INCIDENTS DURING PROCESSING

Red lamp (non labelled) glows

3. This can be caused by one of the following:

(1) Low voltage.

(2) Faulty fuse.

(3) One of the motors not synchronized.

(4) Parity error in HSS.

Incidents on Card Reader/Punch

Read check incident

4. This is indicated by the READ CHECK lamp glowing which signals that a difference has been detected between the check digits calculated at RS1 and RS2 for the same card. The computer stops when the next RAC, IBC or EMP instruction is staticized. The recovery procedure is as follows:

(1) Remove the cards from the hopper.

(2) Clear the track by pressing the CLEAR PRIMARY push-button.

(3) Remove the last five cards from the normal stacker and place the last three of these cards at the front of the card pack.

(4) Replace the card pack in the hopper and press START.

WARNING

> The two remaining cards (first card not checked at RS5, second card not punched) have to be checked, punched and selected as required by the operator.

Punch check incident

5. This is indicated by the PUNCH CHECK lamp glowing which signals that an error has been detected between the check digit calculated at RS2 (which may have been altered by punching) and the check digit calculated at RS5 for the same card. The computer stops when the next RAC, IBC or EMP instruction is staticized. The recovery procedure is as follows:

(1) Remove the last card from stacker S1, this card can be checked at a later stage.

(2) Press START.

Card jam on primary track

6. This incident relates to a card jam between RS1 and the Punching station. It is signalled by the PRIMARY TRACK lamp glowing. The programme will continue until the first RAC, IBC or EMP instruction is staticized. At this point the START lamp glows. The SYNCH lamp glows at the end of the mechanical cycle. The recovery procedure is as follows:

(1) Remove the jammed cards.

(2) Press SYNCH, the lamp incorporated in the push-button glows until the drives are synchronized.

(3) Remake the damaged cards and place them at the head of the pack in the hopper.

WARNING

> The card removed from CS4 must be punched, checked and selected by the operator. The card at CS5 must be checked and selected and the card at CS6 must be selected by the operator.

(4) Press START.

Card jam on secondary track

7. This incident relates to a card jam between CS1' and CS2' on the secondary track. It is signalled by the SEC'Y TRACK lamp glowing. The programme will continue until the first RAC, IBC or EMP instruction is staticized. At this point the START lamp glows. The SYNCH lamp glows at the end of the mechanical cycle. The recovery procedure is as follows:

(1) Remove the cards from the secondary track.

(2) Press SYNCH, the lamp incorporated in the push-button glows until the drives are synchronized.

(3) Press START to continue processing.

Note... cards will be fed onto the secondary track when the 1st IBC instruction occurs.

Card jam on common track

8. This incident relates to a card jam on the common section of the track i.e. from CS5 onwards. It is signalled by the RECEP lamp glowing. The programme will continue until the first RAC, IBC or EMP instruction is staticized. The START lamp glows at this point. The SYNCH lamp glows at the end of the mechanical cycle. The recovery procedure is as follows:

(1) Remove the jammed cards.

(2) Press SYNCH, the lamp incorporated in the push-button glows until the drives are synchronized.

(3) Press START.

(4) The damaged cards must be made and sorted manually.

Card slot jam on primary track

9. This incident occurs when a card fails to reach CS1. The programme continues until the first RAC, IBC or EMP instruction is staticized. The START lamp glows at this point.

10. There are two cases:

(1) The PRIMARY TRACK, FEED 1 CARD and START lamps glow (Hopper not empty). This signals a card slot jam. The recovery procedure is as follows:

(a) Remove the cards from the hopper.

(b) Remove jammed card and remake this card (if required).

(c) Place remade card at the front of the card pack and replace card pack in the hopper.

(d) Press FEED 1 CARD. If the card is fed the lamp incorporated in this push-button is extinguished.

(e) Press START.

(2) Only the PRIMARY TRACK and START lamp glow. This indicates that the hopper is empty. Again two cases are possible:

(a) The job is not finished. Fill the hopper and press START.

(b) The job is finished. Press START this causes the LC socket on the condition plugboard to emit.

Card slot jam on secondary track

11. This incident occurs when a card fails to reach CS1' on the secondary track. It is signalled by the SEC'Y TRACK lamp glowing. The programme will continue until the first RAC, IBC or EMP instruction is staticized. The START lamp glows at this point.

12. There are two cases:

(1) The secondary hopper is empty. Fill the secondary hopper and press START.

(2) There is a card slot jam. Remove the wrecked card and press START.

Note... cards will be fed onto the secondary track when the first IBC instruction occurs.

Full stacker

13. This incident occurs when one of the three stackers is full. It is signalled by the RECEP lamp glowing. The programme blocks on the next RAC, IBC or EMP instruction and the START lamp glows at this point. Empty the appropriate stacker(s) and press START.

Incidents on the Printer

End of paper (or torn paper)

14. Lamp PF1 (or PF2) glows when the end of the sheet is detected. The programme blocks on the first PRI instruction following this condition . The START lamp glows at this point . The operator changes the paper using the same procedure as at the start of a job. Work is restarted by pressing START.

Faulty paper movement (abnormal skip or torn paper)

15. Lamp PF1 (or PF2) glows. The programme blocks on the next PRI instruction (without waiting for the end of sheet condition). The START lamp glows at this point. The procedure for restarting work is as follows:

(1) Position the paper using the same procedures as at the start of a job.

(2) Synchronize the paper with the paper control tape.

(3) Press START.

(4) At the end of the job, rerun the group in which the incident occurred in order to obtain a suitable print-out.

Ribbon fault

16. This is signalled by the RIBBON lamp glowing. The programme blocks on the first PRI instruction. The START lamp glows at this point. The restart procedure is as follows:

(1) Repair fault or insert a new ribbon.

(2) Press START.

Programme Faults

Capacity overflow

17. This is signalled by the CAP. O'FLOW and START lamps glowing. Enter the file and programme level at which the fault occurred on the post mortem sheet and press START.

STOPS DURING PROCESSING

STOP PUSH-BUTTON

18. When the STOP push-button is pressed, the programme stops after the next RAC, IBC, EMP or PRI instruction has been staticized. The programme is restarted from the level on which it was blocked by pressing START.

PROGRAMMED STOP

19. This is signalled by the HALT and START lamps glowing. The operator must follow the instructions given on his instruction sheet.

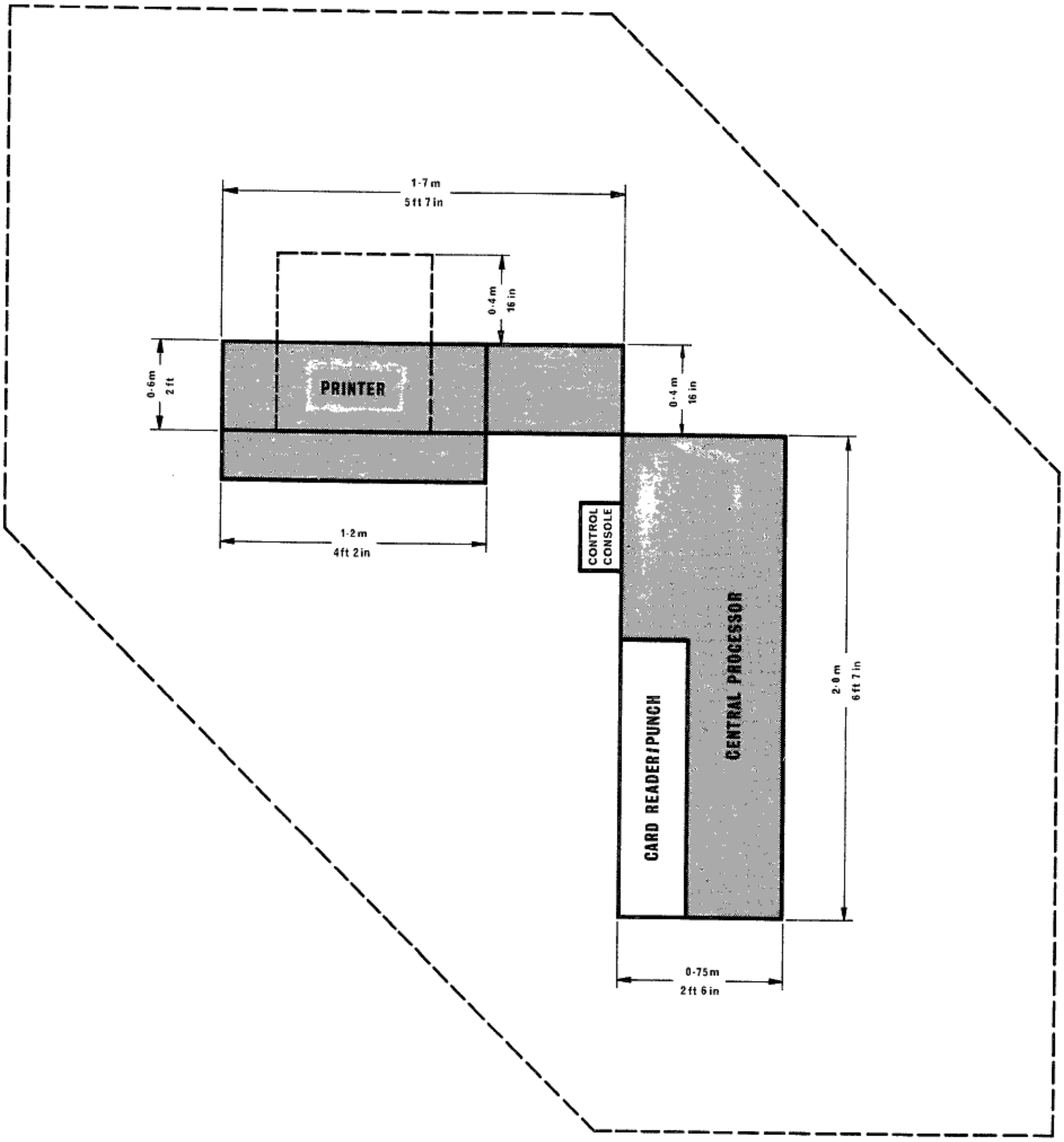
SWITCHING OFF

20. The contents of the HSS will not be altered, if the following procedure is used:

(1) Press STOP.

(2) When the machine stops press POWER OFF

Note... pressing POWER OFF directly will cause the machine to be switched off but there is a risk of altering the HSS content.



WEIGHT: 1300 Kg (2,866 lb)

MAXIMUM HEIGHT: 1·7m (5ft 7in)

POWER CONSUMPTION: 2·25 kW

OVERALL WORKING AREA: 13 sq.m (140 sq.ft)

Physical characteristics of the GAMMA 10

PRINT ORDER	CHARACTER	PUNCH CODE			MACHINE CODE
	SPACE				0
1*	□	9	8	12	61
2*	⅓	9	8	6	46
3*	½	9	8	4	44
4*	Ω	9	7	0	55
5*	"	9	7	3	51
6*	£	9	7	11	30
7*	=	9	8		40
8*	$	9	7	12	62
9*	:	9	7	1	49
10*	)	9	8	1	41
11*	(	9	8	0	47
12*	%	9	7	4	52
13*	+	9	8	3	43
14*	/	9	7		48
15*	&	9	7	5	53
16	'	9	7	2	50
17	*	9	7	6	54
18	12	9		12	60
19	Z	9		6	38
20	Y	9		5	37
21	X	9		4	36
22	W	9		3	35
23	V	9		2	34
24	U	9		1	33
25	T	9		0	39
26	S	9		11	28
27	R		8	6	14
28	Q		8	5	13
29	P		8	4	12

PRINT ORDER	CHARACTER	PUNCH CODE			MACHINE CODE
30	O		7	12	58
3	N		8	3	11
32	M		8	2	10
33	L		8	1	32
34	K		8	0	15
35	J		8	11	25
36	I		8	12	57
37	H		7	6	22
38	G		7	5	21
39	F		7	4	20
40	E		7	3	19
41	D		7	2	18
42	C		7	1	17
43	B		7	0	23
44	A		7	11	26
45	10			11	24
46	11			12	56
47	.	9	8	5	45
48	,	9	8	11	29
49	-	9	8	2	42
50	9	9			9
51	8		8		8
52	7		7		7
53	6			6	6
54	5			5	5
55	4			4	4
56	3			3	3
57	2			2	2
58	1			1	1
59	0			0	16

* Characters which may be replaced by CMC7 numerals and symbols

MARKER	MACHINE CODE						
	BINARY						DECIMAL
	2^5	2^4	2^3	2^2	2^1	2^0	
+ NONSET	0	1	1	0	1	1	27
+ SET	0	1	1	1	1	1	31
- NONSET	1	1	1	0	1	1	59
- SET	1	1	1	1	1	1	63

GAMMA 10 punch and machine codes

APPENDIX 3

INPUT INSTRUCTIONS

Mnemonic	Meaning	O	A	B
RAC	Read a Card	32	00	
INC	Input Numeric Characters	28	Group 0 Area No.	No. of locations
IAC	Input All Characters	29	Group 0 Area No.	No. of locations

OUTPUT INSTRUCTIONS

Mnemonic	Meaning	O	A	B
PRI	Print	34	PF1	PF2
			0 - 3 line-spaces 40 - 46 skips 46 - last line	
SSP	Select Stacker Punch	33		00 - Normal 01 - S1 02 - S2
OAC	Output All Characters	24	Group 0 Area No.	No. of locations
ONS	Output Numeric Suppressed	25	Group 0 Area No.	No. of locations
ONU	Output Numeric Unsuppressed	27	Group 0 Area No.	No. of locations
OSC	Output Single Character	26	Machine Code of Character	No. of locations

INSTRUCTION MODIFICATION

Mnemonic	Meaning	O	A	B
MOD	Modify	31	Group 0 Area No.	No. of O, A, B Parameters

SHIFT INSTRUCTIONS

Mnemonic	Meaning	O	A	B
SHR	Shift Right	22	Group 0 Area No.	No. of Shifts
SHL	Shift Left	23	Group 0 Area No.	No. of Shifts
SCF	Shift Character Fill	18	Machine Code of Character	Group 0 Area No.

TRANSFER INSTRUCTIONS

Mnemonic	Meaning	O	A	B
TRF	Transfer	08	Group 0 Area No.	Group 0 Area No.
TF1	Transfer From Group 1	09	Group 1 Area No.	Group 0 Area No.
TF2	Transfer From Group 2	10	Group 2 Area No.	Group 0 Area No.
TF3	Transfer From Group 3	11	Group 3 Area No.	Group 0 Area No.
TT1	Transfer To Group 1	13	Group 0 Area No.	Group 1 Area No.
TT2	Transfer To Group 2	14	Group 0 Area No.	Group 2 Area No.
TT3	Transfer To Group 3	15	Group 0 Area No.	Group 3 Area No.
TWA	Transfer Within Area	12	Group 0 Area No.	Group 0 Area No.

MISCELLANEOUS INSTRUCTIONS

Mnemonic	Meaning	O	A	B
IBC	Insert Blank Card	32	01	
EMP	Empty	32	02	
HLT	Halt	35		

INSTRUCTION JUMPS

Mnemonic	Meaning	O	A	B
JID	Jump Inst. Dummy	00	00	Destination Inst. No.
JIL	Jump Inst. if result <	00	01	"
JIE	Jump Inst. if result =	00	02	"
JIX	Jump Inst. if result ≤	00	03	"
JIG	Jump Inst. if result >	00	04	"
JII	Jump Inst. if result ≠	00	05	"
JIY	Jump Inst. if result ≥	00	06	"
JIU	Jump Inst. Unconditional	00	07	"
JIS	Jump Inst. if condition SET	06	Marker No.	"
JIN	Jump Inst. if condition NONSET	04	Marker No.	"

SEGMENT JUMPS

Mnemonic	Meaning	O	A	B
JSD	Jump Seg. Dummy	01	00	Destination Seg. No.
JSL	Jump Seg. if result<	01	01	"
JSE	Jump Seg. if result=	01	02	"
JSX	Jump Seg. if result ≤	01	03	"
JSG	Jump Seg. if result >	01	04	"
JSI	Jump Seg. if result ≠	01	05	"
JSY	Jump Seg. if result ≥	01	06	"
JSU	Jump Seg. Unconditional	01	07	"
JSS	Jump Seg. if condition SET	07	Marker No.	"
JSN	Jump Seg. if condition NONSET	05	Marker No.	"

MARKER INSTRUCTIONS

Mnemonic	Meaning	O	A	B
TCC	Transfer Condition to Condition	36	Marker No.	Marker No.
ASC	Allocate Sign according to Condition	37	Marker No.	Marker No.
ACS	Allocate Condition according to Sign	38	Marker No.	Marker No.
TSS	Transfer Sign to Sign	39	Marker No.	Marker No.

ARITHMETIC INSTRUCTIONS

Mnemonic	Meaning	O	A	B
ADD	Add	16	Addend Area No.	Augend Area No.
SUB	Subtract	17	Subtrahend Area No.	Minuend Area No.
MUL	Multiply	20	Multiplicand Area No.	Multiplier Area No.
DIV	Divide	21	Divisor Area No.	Quotient Area No.
CMP	Compare	19	Area No. (first operand)	Area No. (Second operand)

RETURN FROM SUBROUTINE

Mnemonic	Meaning	O	A	B
SRA	Store Return Address	30	Source SAR No.	Receiving SAR No
JBK	Jump Back	03		SAR No.

Gamma 10 instruction codes

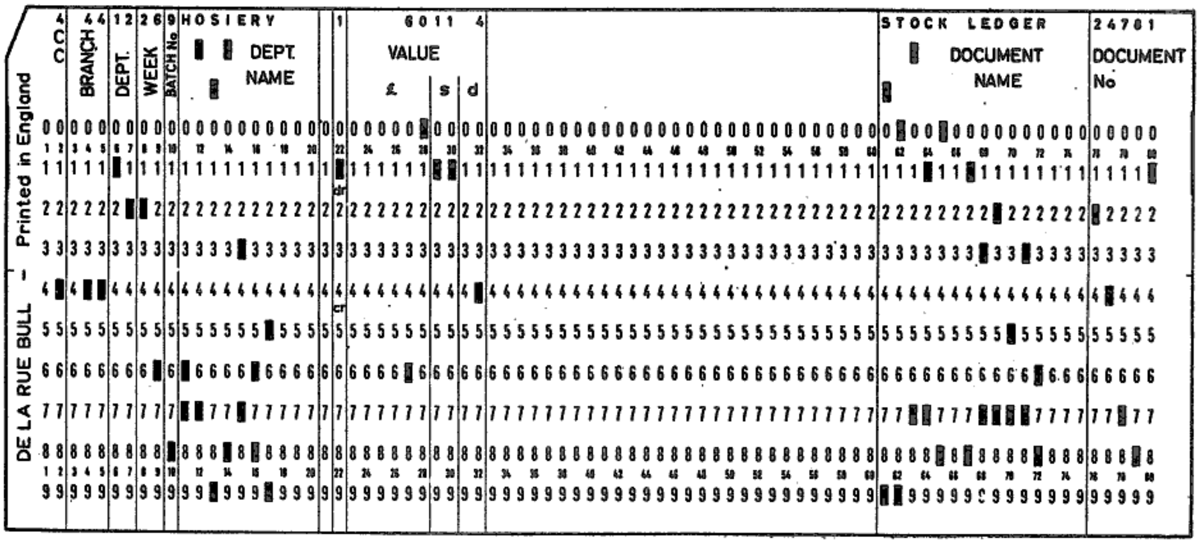
APPENDIX 4

HSS Capacity		1024 Locations	2048 Locations		4096 Locations				
Groups used		Group 0	Group 0	Groups 0 and 1	Group 0	Groups 0 and 1	Groups 0, 1 and 2	Groups 0, 1, 2 and 3	
Registers	SAR	128	128	128	128	128	128	128	Fixed
	AAR - Group 0	128	128	128	128	128	128	128	
	AAR Group 1, 2, 3	0	0	128	0	128	256	384	
Servicing		24	24	24	24	24	24	24	
MARKERS	Group 0 Areas	64	64	64	64	64	64	64	
	Last Area Marker	1	1	1	1	1	1	1	
	Group 1	0	0	64	0	64	64	64	Variable
	Group 2	0	0	0	0	0	64	64	
	Group 3	0	0	0	0	0	0	64	
Total of locations used by Markers, Registers, etc.		345	345	537	345	537	729	921	
Locations Available for Data and Programme		675	1703	1511	3751	3559	3367	3175	

Note: It is assumed that all areas in every Group have been allocated

DATA DISTRIBUTION IN THE HIGH SPEED STORE

APPENDIX 5 (I)

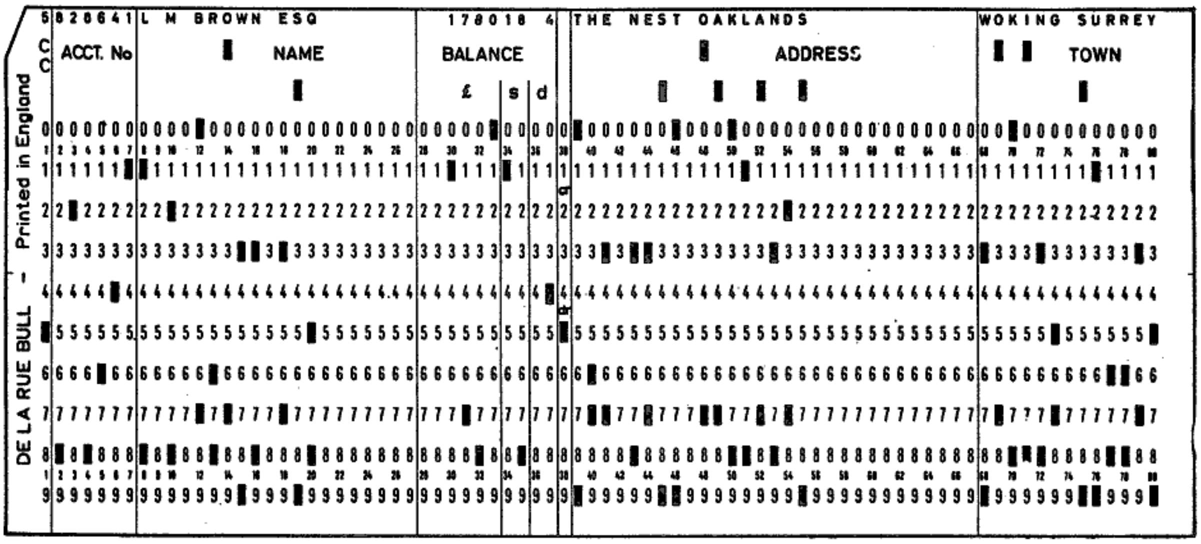


MNEMONIC	A	B	OPERATION
RAC	00		Transfers the content of the Condition Buffer to markers 1 to 32 and causes a card to be read
INC	15	05	Copies the content of columns 76 to 80 into Group 0 area 15.
IAC	35	43	Copies the content of columns 61 to 75 into Group 0 area 35 and eliminates columns 33 to 60.
INC	26	12	Copies the content of columns 23 to 32 into Group 0 area 26 and eliminates columns 21 and 22.
IAC	28	10	Copies the content of columns 11 to 20 into Group 0 area 28.
INC	16	01	Copies the content of column 10 into Group 0 area 16.
INC	17	02	Copies the content of columns 8 and 9 into Group 0 area 17.
INC	18	02	Copies the content of columns 6 and 7 into Group 0 area 18.
INC	19	05	Copies the content of columns 3 to 5 into Group 0 area 19 and eliminates columns 1 and 2.

Note... columns 21 and 33 to 60 have been eliminated because they are not used. Columns 1 and 2 and 22 have been eliminated because they contain qualitative data detected via the Condition Plugboard and stored in markers.

TYPICAL PROGRAMME INPUT SEQUENCE

APPENDIX 5 (II)



MNEMONIC	A	B	OPERATION
RAC	00		
IAC	00	43	Eliminates content of columns 38 to 80 by attempting to copy this data into Group 0 area 00 which has length 0. Note... content of column 38 contains qualitative data detected via the Condition Plugboard and stored in a marker.
INC	14	10	Copies the content of columns 28 to 37 into Group 0 area 14.
IAC	28	20	Copies the content of columns 8 to 27 into Group 0 area 28.
INC	22	07	Copies the content of columns 2 to 7 into Group 0 area 22 and eliminates column 1.

Note... column 1 contains qualitative data detected via the Condition Plugboard and stored in a marker.

TYPICAL PROGRAMME INPUT SEQUENCE

Area 22

8	2	8	6	4	1

Area 28

L		M		B	R	O	W	N		E	S	Q							

Area 14

1	7	8	0	1	8		4		

Area 48

T	H	E		N	E	S	T		O	A	K	L	A	N	D	S												

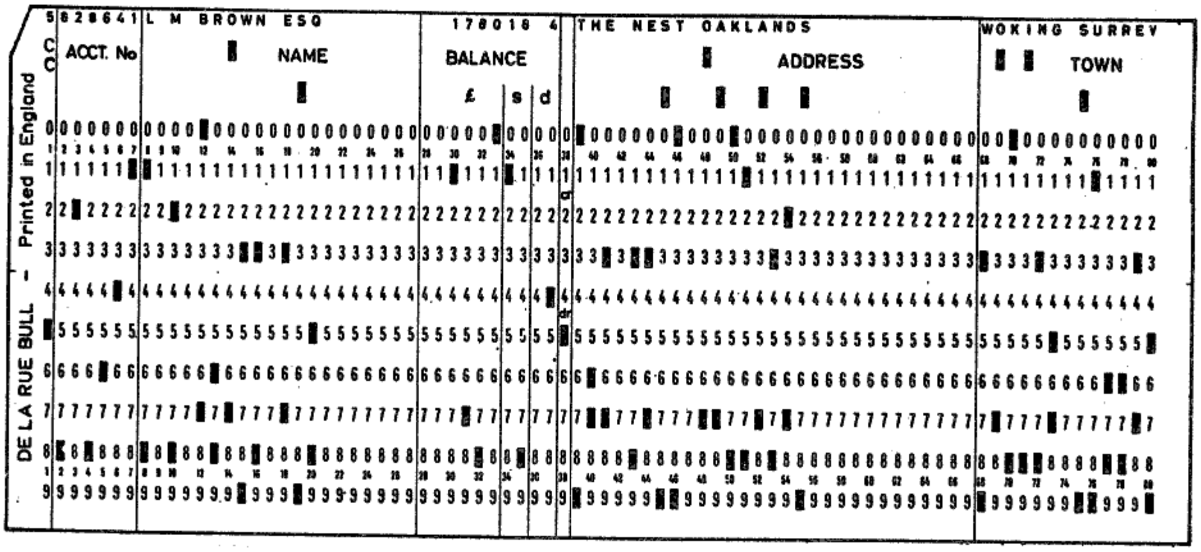
Area 49

W	O	K	I	N	G		S	U	R	R	E	Y

5/3	5 if Credit Balance 3 if Debit Balance

MNEMONIC	A	B	OPERATION
SSP		01	Commands card selection into stacker S1 and informs the computer that the following instructions apply to punched output.
OAC	49	13	Copies the content of Group 0 area 49 into Punch Buffer locations 80 to 68.
OAC	48	29	Copies the content of Group 0 area 48 into Punch Buffer locations 67 to 39.
ONU	50	01	Copies the content of Group 0 area 50 into Punch Buffer location 38.
ONS	14	10	Copies the content of Group 0 area 14 into Punch Buffer locations 37 to 28.
OAC	28	20	Copies the content of Group 0 area 28 into Punch Buffer locations 27 to 8.
ONU	22	06	Copies the content of Group 0 area 22 into Punch Buffer locations 7 to 2.
OSC		01	Inserts the appropriate card code (machine code given in A of the instruction) into Punch Buffer location 1.

APPENDIX 6 (I.b)



TYPICAL PROGRAMME OUTPUT SEQUENCE FOR PUNCHING

Area 24

cc

Area 28

7	0	1	2	3	2

Area 30

D	U	S	T	B	I	N		G		I									

Area 32

1	0	0	0

Area 34

G	O	O	D		W	A	R	E	S		L	T	D											

Area 36

7	3	2	5	2	7

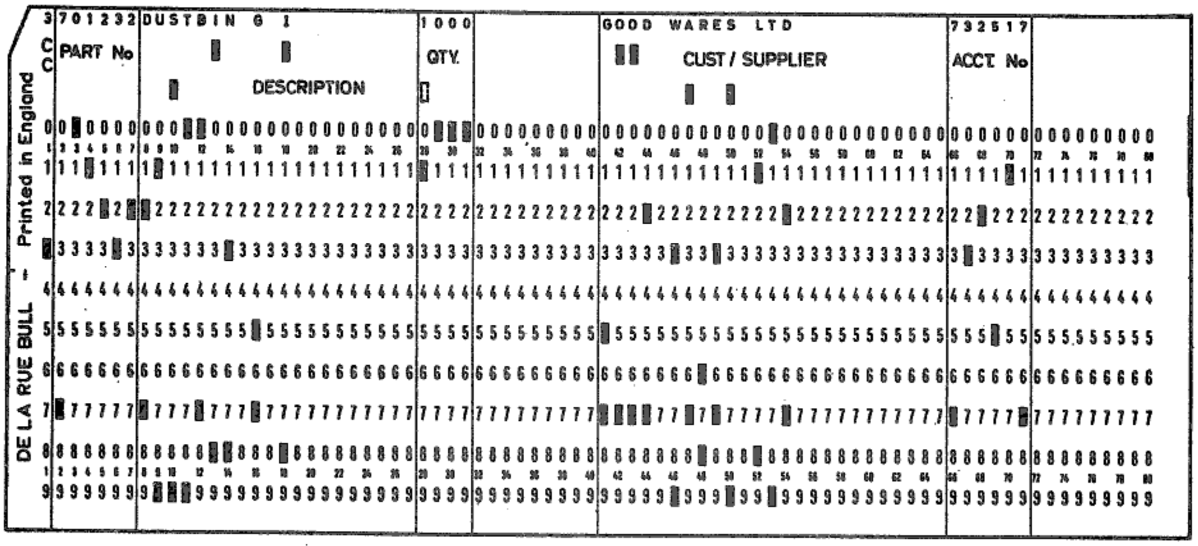
MNEMONIC	A	B	OPERATION
SSP		00	Commands card selection into the Normal stacker and informs the computer that the following instructions apply to punched output.
OAC	00	09	Skips columns 80 to 72 of the Punch Buffer.
ONU	36	06	Copies the content of Group 0 area 36 into Punch Buffer locations 71 to 66.
OAC	34	34	Copies the content of Group 0 area 34 into Punch Buffer locations 65 to 41 and skips columns 40 to 32.
ONU	32	04	Copies the content of Group 0 area 32 into Punch Buffer locations 31 to 28.
OAC	30	20	Copies the content of Group 0 area 30 into Punch Buffer locations 27 to 8.
ONU	28	06	Copies the content of Group 0 area 28 into Punch Buffer locations 7 to 2.
ONU	24	01	Copies the content of Group 0 area 24 into Punch Buffer location 1.

The above sequence covers the full eighty columns but has not output any overpunches. The following two instructions must be programmed for the 11 overpunch in column 28.

OSC	00	52	Skips Punch Buffer locations 80 to 29.
OSC	56	28	Inserts the 11 overpunch in Punch Buffer location 28 and skips locations 27 to 1.

APPENDIX 6 (IIb)

Note... this second sequence again covers the full eighty locations of the Punch Buffer.



TYPICAL PROGRAMME OUTPUT SEQUENCE FOR PUNCHING

Area 14

0	2	7	8	9	0

Area 16

		R	A	C	K		&		P	I	N	I	O	N		A	S	S	Y

Area 18

2	1	8	0

Area 20

	8		1	8			4

Area 22

		1	9	4	3	8		1	2		8

MNEMONIC	A	B	OPERATION
PRI	02	00	Commands a movement of 2 line-spaces on **PF1 and informs** the computer that the following instructions apply to printed output.
ONU	00	53	Skips Print Buffer locations 120 to 68.
OAC	22	15	Copies the content of Group 0 area 22 into Print Buffer locations 67 to 56 and skips locations 55 to 53.
OAC	20	10	Copies the content of Group 0 area 20 into Print Buffer locations 52 to 45 and skips locations 44 and 43.
ONS	18	07	Copies the content of Group 0 area 18 into Print Buffer locations 42 to 39 and skips locations 38 to 36
OAC	16	22	Copies the content of Group 0 area 16 into Print Buffer locations 35 to 16 and skips locations 15 and 14.
OAC	14	13	Copies the content of Group 0 area 14 into Print Buffer locations 13 to 8 and skips locations 7 to 1.

STOCK VALUATION

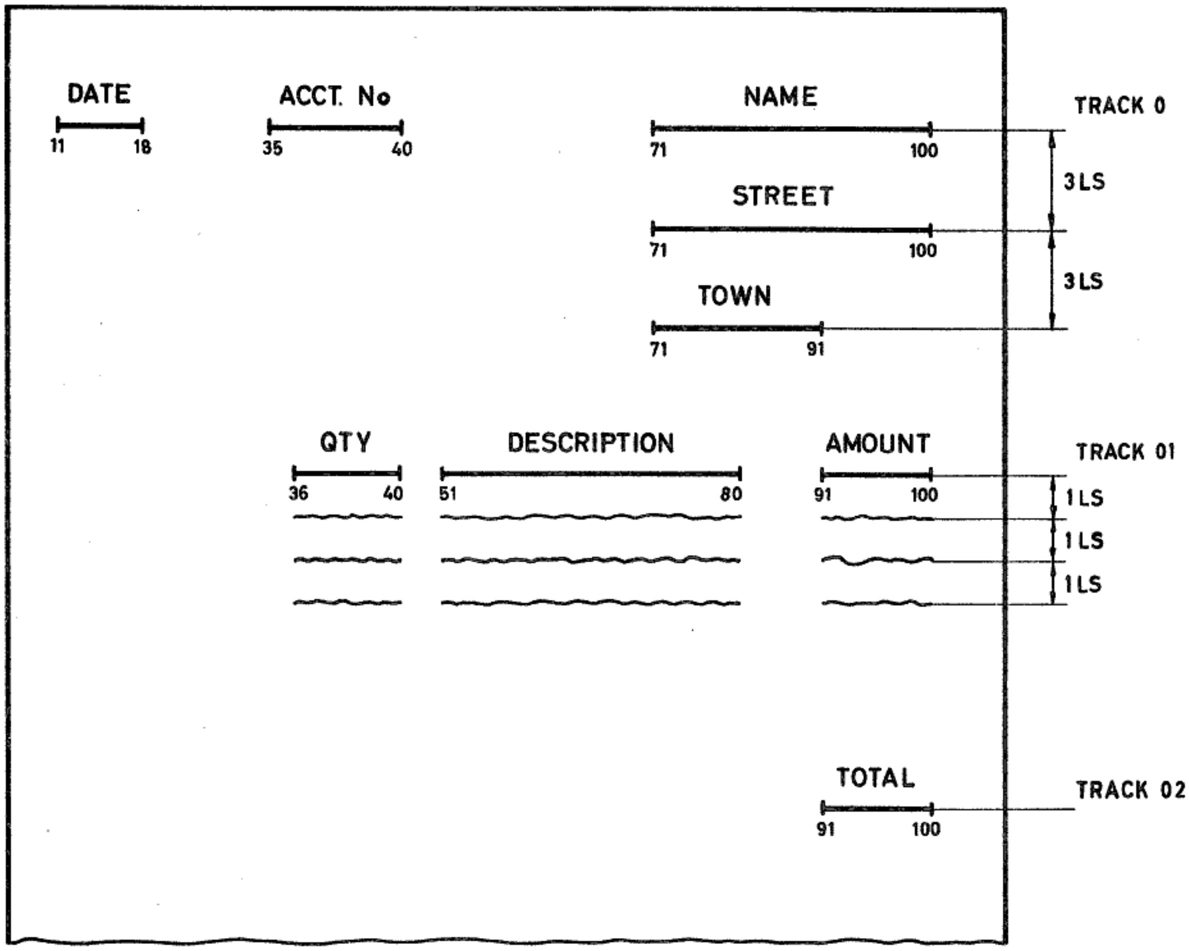
	PART No	DESCRIPTION	STOCK	UNIT PRICE £ s d	TOTAL £ s d	
1	8 13	16 35	39 42	45 52	56 67	120
	0 2 7 8 9 0	RACK & PINION ASSY	2 1 8 0	8 18 4	19438 12 8	

TYPICAL PROGRAMME OUTPUT SEQUENCE FOR PRINTING

AREA No.	LENGTH	CONTENT
12	5	Quantity
14	30	Description
16	6	Account Number
18	8	Date
20	30	Name
22	30	Street
24	20	Town
26	10	Amount
28	10	Total
30	1	Determined by SCF instruction

MNEMONIC	A	B	OPERATION
⋮			
PRI	40		Commands printing, the paper movement is controlled by track 0 of the paper control tape.
OAC	00	20	Skips Print buffer locations 120 to 101.
OAC	20	60	Copies the content of Group 0 area 20 into Print Buffer locations 100 to 71 and skips locations 70 to 41.
OAC	16	22	Copies the content of Group 0 area 16 into Print Buffer locations 40 to 35 and skips locations 34 to 19.
ONS	18	18	Copies the content of Group 0 area 18 into Print Buffer locations 18 to 11 and skips locations 10 to 1.
⋮			
PRI	03		Commands printing - 3 line-spaces.
OAC	00	20	Skips Print Buffer locations 120 to 101.
OAC	22	60	Copies the content of Group 0 area 22 into Print Buffer locations 100 to 71 and skips locations 70 to 41.
OAC	00	40	Skips locations 40 to 01 in the Print Buffer.
⋮			

PRI	03		Commands printing - 3 line-spaces.
OAC	00	30	Skips Print Buffer locations 120 to 91.
OAC	24	63	Copies the content of Group 0 area 24 into Print Buffer locations 90 to 71 and skips locations 70 to 28.
OAC	00	27	Skips Print Buffer locations 27 to 01.
⋮			
SCF	41	30	Inserts machine code 41 into area 30.
→ ⋮			
MOD	30	01	Modifies the A character of the next instruction i. e. paper movement controlled by track 1 or commanded for 1 line-space.
PRI			A character modified by preceding MOD instruction.
OAC	00	20	Skips Print Buffer locations 120 to 101.
ONU	26	20	Copies the content of Group 0 area 26 into Print Buffer locations 100 to 91 and skips locations 90 to 81.
OAC	14	40	Copies the content of Group 0 area 14 into Print Buffer locations 80 to 51 and skips locations 50 to 41.
OAC	12	40	Copies the content of Group 0 area 12 into Print Buffer locations 40 to 36 and skips locations 35 to 01.
SCF	01	30	Inserts 1 into area 30.
JIS	45		Marker 45 will be NONSET if a change of group occurs.
⋮			
PRI	42		Commands printing, the paper movement is controlled by track 2 of the paper control tape.
OAC	00	20	Skips Print Buffer locations 120 to 101.
OAC	28	10	Copies the content of Group 0 area 28 into Print Buffer locations 100 to 91.
OSC	30	60	Inserts the £ symbol into Print Buffer location 90 and skips locations 89 to 31.
OAC	00	30	Skips Print Buffer locations 30 to 01.
⋮			



TYPICAL PROGRAMME OUTPUT SEQUENCE FOR PRINTING

APPENDIX 8 (a)

HEADER CARD			
Columns 65 and 66	-	Punch 04 if only Group 0 used	
	-	Punch 06 if Group 0 and Group 1 used	
	-	Punch 08 if Groups 0, 1, and 2 used	
	-	Punch 10 if Groups 0, 1, 2 and 3 used	
Column 67	-	Punch 11 if checking	Usually punched by Card Pack Duplication Routine
	-	Punch 12 if loading	
Columns 68 to 71	-	Punch 11 in each column	
Columns 74 to 77	-	Programme Identification (optional)	

SEGMENT CARD			
Column 67	-	Punch 11 if checking	Usually punched by Card Pack Duplication Routine
	-	Punch 12 if loading	
Columns 68 and 69	-	Segment Number	
Columns 70 and 71	-	Punch 11 in each column	
Column 72	-	Punch 1 if blank segment	
Columns 74 to 77	-	Programme Identification (optional)	

INSTRUCTION CARD			
Columns 54 to 56	-	Mnemonic	
Columns 58 and 59	-	O (Decimal code)	
Columns 60 and 61	-	A (decimal)	
Columns 62 and 63	-	B (decimal)	
Column 67	-	Punch 11 if checking	Usually punched by Card Pack Duplication Routine
	-	Punch 12 if loading	
Columns 68 and 69	-	Segment Number	
Columns 70 and 71	-	Instruction Number	
Column 72	-	Punch 1 if last instruction of segment	
Column 73	-	Punch 1 if last instruction of programme	
Columns 74 to 77	-	Programme Identification (optional)	

GROUP CARD

Column 67	-	Punch 11 if checking	Usually punched by Card Pack Duplication Routine
	-	Punch 12 if loading	
Column 68	-	Punch 9	
Column 69	-	Group number i.e. 0, 1, 2 or 3	
Columns 70 and 71	-	Punch 11 in each column	
Columns 74 to 77	-	Programme Identification (optional)	

AREA CARD

Columns 1 to 63	-	Data to be stored - column 63 corresponds to location 1 (i.e. rightmost location) of the area	
Column 64	-	Punch 4 to SET marker	
	-	Punch 9 to insert negative sign	
Columns 65 and 66	-	Length of area	
Column 67	-	Punch 11 if checking	Usually punched by Card Pack Duplication Routine
	-	Punch 12 if loading	
Column 68	-	Punch 9	
Column 69	-	Group number i.e. 0, 1, 2 or 3	
Columns 70 and 71	-	Area number	
Column 72	-	Punch 1 if last area in group	
Column 73	-	Punch 1 if final area	

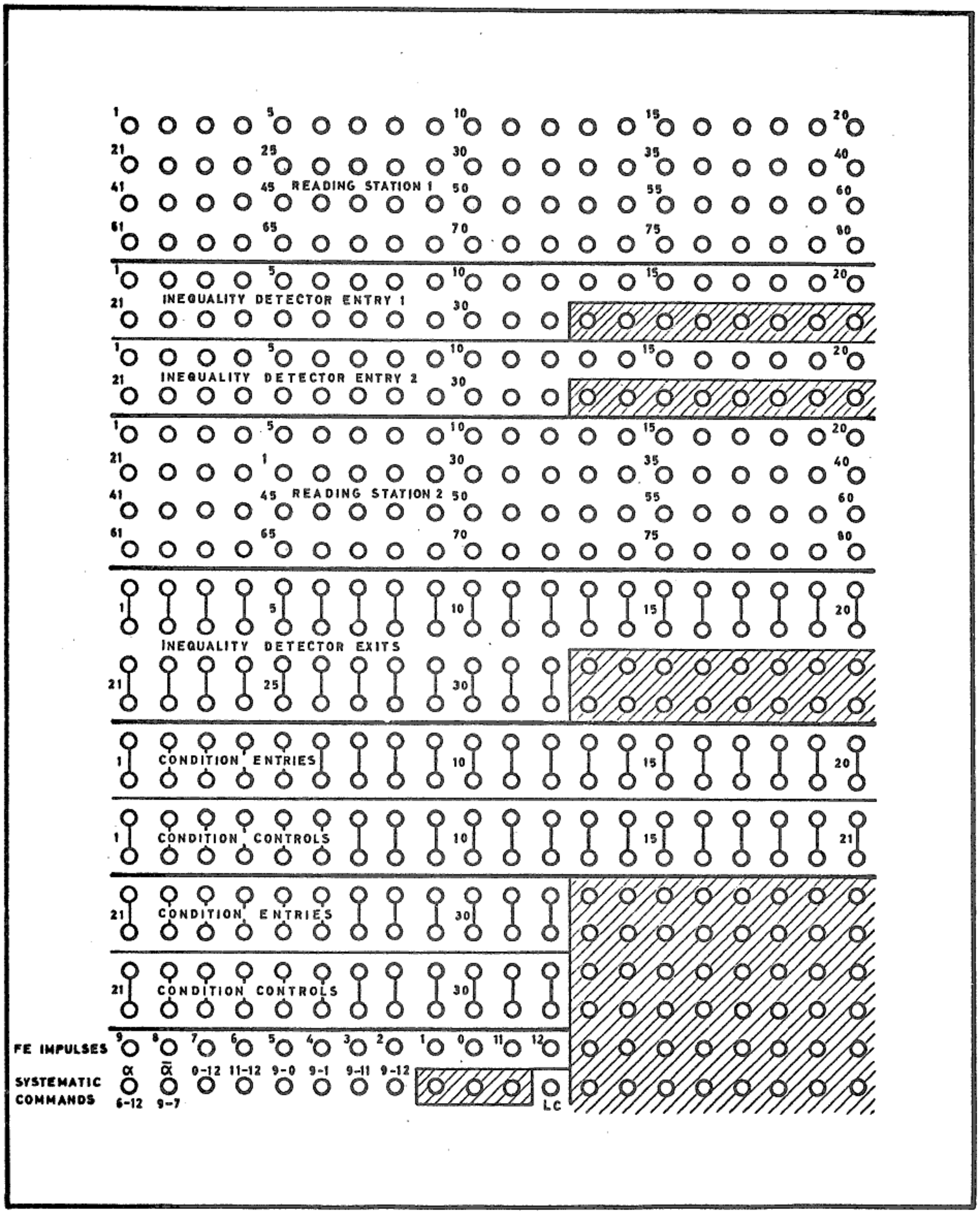
DATA FILL (MARKER) CARD

Column 64	-	Punch 4 to SET the marker. The marker will be NON-SET if this column is blank
Columns 65 and 66	-	Marker number
Column 68	-	Punch 4

DATA FILL (AREA) CARD

Columns 1 to 63	-	Data to be stored - column 63 corresponds to location 1 (i.e. rightmost location) of the area
Column 64	-	Punch 9 to insert negative sign. The area will be positive if this column is blank
Columns 65 and 66	-	Area number
Column 68	-	Punch 9

CARD FORMATS



CONDITION PLUGBOARD